Deeper Inside PageRank

Amy N. Langville† and Carl D. Meyer*

October 20, 2004

Abstract

This paper serves as a companion or extension to the "Inside PageRank" paper by Bianchini et al. [19]. It is a comprehensive survey of all issues associated with PageRank, covering the basic PageRank model, available and recommended solution methods, storage issues, existence, uniqueness, and convergence properties, possible alterations to the basic model, suggested alternatives to the traditional solution methods, sensitivity and conditioning, and finally the updating problem. We introduce a few new results, provide an extensive reference list, and speculate about exciting areas of future research.

Key words: Markov chains, power method, convergence, stationary vector, PageRank, updating

† Department of Mathematics,
N. Carolina State University,
Raleigh, NC 27695-8205, USA
anlangvi@unity.ncsu.edu
Phone: (919) 513-2111,
Fax: (919) 513-3798

* Department of Mathematics,
Center for Research in Scientific Computation,
N. Carolina State University, Raleigh, N.C. 27695-8205, USA
meyer@math.ncsu.edu
Phone: (919) 515-2384,
Fax: (919) 515-3798
Research supported in part by NSF CCR-ITR-0113121 and NSF CCR-0318575.

1 Introduction

Many of today's search engines use a two-step process to retrieve pages related to a user's query. In the first step, traditional text processing is done to find all documents using the query terms, or related to the query terms by semantic meaning. This can be done by a lookup into an inverted file, with a vector space method, or with a query expander that uses a thesaurus. With the massive size of the

Web, this first step can result in thousands of retrieved pages related to the query. To make this list manageable for a user, many search engines sort this list by some ranking criterion. One popular way to create this ranking is to exploit the additional information inherent in the Web due to its hyperlinking structure. Thus, link analysis has become the means to ranking. One successful and well-publicized link-based ranking system is PageRank, the ranking system used by the Google search engine. Actually, for pages related to a query, an IR (information retrieval) score is combined with a PR (PageRank) score to determine an overall score, which is then used to rank the retrieved pages [20]. This paper focuses solely on the PR score.

We begin the paper with a review of the most basic PageRank model for determining the importance of a webpage. This basic model, so simple and so elegant, works well, but part of the model's beauty and attraction lies in its seemingly endless capacity for "tinkering". Some such tinkerings have been proposed and tested. In this paper, we explore these previously suggested tinkerings to the basic PageRank model and add a few more suggestions and connections of our own. For example, why has the PageRank convex combination scaling parameter traditionally been set to .85? One answer, presented in section 5.1, concerns convergence to the solution. However, we provide another answer to this question in section 7 by considering the sensitivity of the problem. Another area of fiddling is the uniform matrix $\mathbf{E}$ added to the hyperlinking Markov matrix $\mathbf{P}$. What other alternatives to this uniform matrix exist? In section 6.3, we present the common answer, followed by an analysis of our alternative answer. We also delve deeper into the PageRank model, discussing convergence in section 5.1.1, sensitivity, stability, and conditioning in section 7, and updating in section 8. The numerous alterations to and intricacies of the basic PageRank model presented in this paper give an appreciation of the model's beauty and usefulness, and hopefully, will inspire future and greater improvements.

2 The Scene in 1998

The year 1998 was a busy year for link analysis models. On the East Coast, a young scientist named Jon Kleinberg, an assistant professor in his second year at Cornell University, was working on a Web search engine project called HITS. His algorithm used the hyperlink structure of the Web to improve search engine results, an innovative idea at the time, as most search engines used only textual content to return relevant documents. He presented his work [74], begun a year earlier at IBM, in January 1998 at the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms held in San Francisco, California. Very nearby, at Stanford University, two Ph.D. candidates were working late nights on a similar project called PageRank. Sergey Brin and Larry Page, both computer science students, had been collaborating on their Web search engine since 1995. By 1998, things were really starting to accelerate for these two scientists. They were using their dorm rooms as offices for the fledgling business, which later became the giant Google. By August 1998, both Brin and Page took a leave of absence from Stanford in order to focus on their growing business. In a public presentation at the Seventh International World Wide Web conference (WWW98) in Brisbane, Australia, their paper "The PageRank citation ranking: Bringing order to the Web" [27] made small ripples in the information science community that quickly turned into waves. The connections between the two models are striking (see [78]) and it's hard to say whether HITS influenced PageRank, or vice versa, or whether both developed independently. Nevertheless, since that eventful year, PageRank has emerged as the dominant link analysis model, partly due to its query-independence, its virtual immunity to spamming and Google's huge business success. Kleinberg was already making a name for himself as an innovative academic, and unlike Brin and Page, did not try to develop HITS into a company. However, later entrepreneurs did; the search engine Teoma uses an extension of the HITS algorithm as the basis of its underlying technology [107]. As a side note, Google kept Brin and Page busy and wealthy enough to remain on leave from Stanford. This paper picks up after their well-cited original 1998 paper and explores the numerous suggestions that have been made to the basic PageRank model, thus, taking the reader deeper inside PageRank. We note that this paper

describes methods invented by Brin and Page, which were later implemented into their search engine Google. Of course, it is impossible to surmise the details of Google's implementation since the publicly disseminated details of the 1998 papers [25, 26, 27]. Nevertheless, we do know that PageRank remains "the heart of [Google's] software ... and continues to provide the basis for all of [their] web search tools", as cited directly from the Google webpage, `http://www.google.com/technology/index.html`.

3 The Basic PageRank model

The original Brin and Page model for PageRank uses the hyperlink structure of the Web to build a Markov chain with a primitive[1] transition probability matrix $\mathbf{P}$. The irreducibility of the chain guarantees that the long-run stationary vector $\boldsymbol{\pi}^T$, known as the PageRank vector, exists. It is well-known that the power method applied to a primitive matrix will converge to this stationary vector. Further, the convergence rate of the power method is determined by the magnitude of the subdominant eigenvalue of the transition rate matrix [108].

3.1 The Markov model of the Web

We begin by showing how Brin and Page, the founders of the PageRank model, force the transition probability matrix, which is built from the hyperlink structure of the Web, to be stochastic and primitive. Consider the hyperlink structure of the Web as a directed graph. The nodes of this digraph represent webpages and the directed arcs represent hyperlinks. For example, consider the small document collection consisting of 6 webpages linked as in Figure 1.

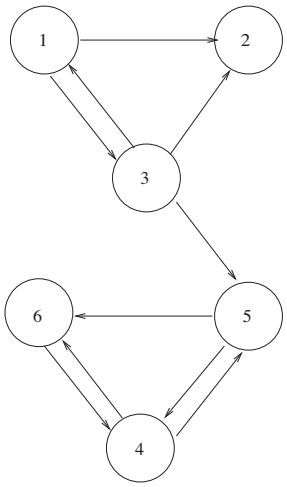

Figure 1: Directed graph representing web of 6 pages

The Markov model represents this graph with a square matrix $\mathbf{P}$ whose element p_{ij} is the probability of moving from state i (page i) to state j (page j) in one time step. For example, assume that, starting

[1] A matrix is *irreducible* if its graph shows that every node is reachable from every other node. A nonnegative, irreducible matrix is *primitive* if it has only one eigenvalue on its spectral circle. An irreducible Markov chain with a primitive transition matrix is called an aperiodic chain. Frobenius discovered a simple test for primitivity: the matrix $\mathbf{A} \geq 0$ is primitive if and only if $\mathbf{A}^m > 0$ for some $m > 0$ [89]. This test is useful in determining whether the power method applied to a matrix will converge.

from any node (webpage), it is equally likely to follow any of the outgoing links to arrive at another node. Thus,

$$\mathbf{P} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}.$$

Any suitable probability distribution may be used across the rows. For example, if web usage logs show that a random surfer accessing page 1 is twice as likely to jump to page 2 as he or she is to jump to page 3, then the first row of $\mathbf{P}$, denoted $\mathbf{p}_1^T$, becomes

$$\mathbf{p}_1^T = (\,0 \quad 2/3 \quad 1/3 \quad 0 \quad 0 \quad 0\,).$$

(Similarly, column i of $\mathbf{P}$ is denoted $\mathbf{p}_i$.) Another weighting scheme is proposed in [9]. One problem with solely using the Web's hyperlink structure to build the Markov matrix is apparent. Some rows of the matrix, such as row 2 in our example above, contain all zeros. Thus, $\mathbf{P}$ is not stochastic. This occurs whenever a node contains no outlinks; many such nodes exist on the Web. Such nodes are called dangling nodes. One remedy is to replace all zero rows, $\mathbf{0}^T$, with $\frac{1}{n}\mathbf{e}^T$, where $\mathbf{e}^T$ is the row vector of all ones and n is the order of the matrix. The revised transition probability matrix called $\bar{\mathbf{P}}$ is

$$\bar{\mathbf{P}} = \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

(We note that the uniform vector $\frac{1}{n}\mathbf{e}^T$ can be replaced with a general probability vector $\mathbf{v}^T > 0$. See section 6.2 for more details about this *personalization vector* $\mathbf{v}^T$.) However, this adjustment alone is not enough to insure the existence of the stationary vector of the chain, i.e., the PageRank vector. Were the chain irreducible, the PageRank vector is guaranteed to exist. By its very nature, with probability 1, the Web unaltered creates a reducible Markov chain. (In terms of graph theory, the web graphs are non-bipartite and not necessarily strongly connected.) Thus, one more adjustment, to make $\mathbf{P}$ irreducible, is implemented. The revised stochastic and irreducible matrix $\bar{\bar{\mathbf{P}}}$ is

$$\bar{\bar{\mathbf{P}}} = \alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{e}\mathbf{e}^T/n = \begin{pmatrix} 1/60 & 7/15 & 7/15 & 1/60 & 1/60 & 1/60 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 19/60 & 19/60 & 1/60 & 1/60 & 19/60 & 1/60 \\ 1/60 & 1/60 & 1/60 & 1/60 & 7/15 & 7/15 \\ 1/60 & 1/60 & 1/60 & 7/15 & 1/60 & 7/15 \\ 1/60 & 1/60 & 1/60 & 11/12 & 1/60 & 1/60 \end{pmatrix}$$

where $0 \leq \alpha \leq 1$ and $\mathbf{E} = 1/n\mathbf{e}\mathbf{e}^T$. This convex combination of the stochastic matrix $\bar{\mathbf{P}}$ and a stochastic perturbation matrix $\mathbf{E}$ insures that $\bar{\bar{\mathbf{P}}}$ is both stochastic and irreducible. Every node is now directly connected to every other node, making the chain irreducible by definition. Although the probability of transitioning may be very small in some cases, it is always nonzero. The irreducibility adjustment also insures that $\bar{\bar{\mathbf{P}}}$ is primitive, which implies that the power method will converge to the stationary PageRank vector $\boldsymbol{\pi}^T$.

4 Storage Issues

The size of the Markov matrix makes storage issues nontrivial. In this section, we provide a brief discussion of more detailed storage issues for implementation. The 1998 paper by Brin and Page [26] and more recent papers by Google engineers [13, 56] provide detailed discussions of the many storage schemes used by the Google search engine for all parts of its information retrieval system. The excellent survey paper by Arasu et al. [6] also provides a section on storage schemes needed by a Web search engine. Since this paper is mathematically-oriented, we focus only on the storage of the mathematical components, the matrices and vectors, used in the PageRank part of the Google system.

For subsets of the web, the transition matrix $\mathbf{P}$ (or its graph) may or may not fit in main memory. For small subsets of the web, when $\mathbf{P}$ fits in main memory, computation of the PageRank vector can be implemented in the usual fashion. However, when the $\mathbf{P}$ matrix does not fit in main memory, researchers must be more creative in their storage and implementation of the essential components of the PageRank algorithm. When a large matrix exceeds a machine's memory, researchers usually try one of two things: they compress the data needed so that the compressed representation fits in main memory and then creatively implement a modified version of PageRank on this compressed representation, or they keep the data in its uncompressed form and develop I/O-efficient implementations of the computations that must take place on the large, uncompressed data.

For modest web graphs for which the transition matrix $\mathbf{P}$ can be stored in main memory, compression of the data is not essential, however, some storage techniques should still be employed to reduce the work involved at each iteration. For example, the $\mathbf{P}$ matrix is decomposed into the product of the inverse of the diagonal matrix $\mathbf{D}$ holding outdegrees of the nodes and the adjacency matrix $\mathbf{G}$ of 0's and 1's is useful in saving storage and reducing work at each power iteration. The decomposition $\mathbf{P} = \mathbf{D}^{-1}\mathbf{G}$ is used to reduce the number of multiplications required in each $\mathbf{x}^T\mathbf{P}$ vector-matrix multiplication needed by the power method. Without the $\mathbf{P} = \mathbf{D}^{-1}\mathbf{G}$ decomposition, this requires $nnz(\mathbf{P})$ multiplications and $nnz(\mathbf{P})$ additions, where $nnz(\mathbf{P})$ is the number of nonzeros in $\mathbf{P}$. Using the vector $diag(\mathbf{D}^{-1})$, $\mathbf{x}^T\mathbf{P}$ can be accomplished as $\mathbf{x}^T\mathbf{D}^{-1}\mathbf{G} = (\mathbf{x}^T).*(diag(\mathbf{D}^{-1}))\mathbf{G}$, where $.*$ represents componentwise multiplication of the elements in the two vectors. The first part, $(\mathbf{x}^T).*(diag(\mathbf{D}^{-1}))$ requires n multiplications. Since $\mathbf{G}$ is an adjacency matrix, $(\mathbf{x}^T).*(diag(\mathbf{D}^{-1}))\mathbf{G}$ now requires an additional $nnz(\mathbf{P})$ additions for a total savings of $nnz(\mathbf{P}) - n$ multiplications. In addition, for large matrices compact storage schemes [12], such as compressed row storage or compressed column storage, are often used. Of course, each compressed format, while saving some storage, requires a bit more overhead for matrix operations.

Rather than storing the full matrix or a compressed version of the matrix, Web-sized implementations of the PageRank model store the $\mathbf{P}$ or $\mathbf{G}$ matrix in an adjacency list of the columns of the matrix [100]. In order to compute the PageRank vector, the PageRank power method (defined in section 5.1) requires vector-matrix multiplications of $\mathbf{x}^{(k-1)T}\mathbf{P}$ at each iteration k. Therefore, quick access to the columns of the matrix $\mathbf{P}$ (or $\mathbf{G}$) is essential to algorithm speed. Column i contains the inlink information for page i, which, for the PageRank system of ranking webpages, is more important than the outlink information contained in the rows of $\mathbf{P}$ or $\mathbf{G}$. For the tiny 6-node web from section 3, an adjacency list representation of the columns of $\mathbf{G}$ is:

Node	Inlinks from
1	3
2	1, 3
3	1
4	5, 6
5	3, 4
6	4, 5

Exercise 2.24 of Cleve Moler's recent book [93] gives one possible implementation of the power method

applied to an adjacency list, along with sample Matlab code. When the adjacency list does not fit in main memory, references [100, 102] suggest methods for compressing the data. References [31, 58] take the other approach and suggest I/O-efficient implementations of PageRank. Since the PageRank vector itself is large and completely dense, containing over 4.3 billion pages, and must be consulted in order to process each user query, Haveliwala [59] has suggested a technique to compress the PageRank vector. This encoding of the PageRank vector hopes to keep the ranking information cached in main memory, thus speeding query processing.

Because of their potential and promise, we briefly discuss two methods for compressing the information in an adjacency list, the gap technique [15] and the reference encoding technique [101, 102]. The gap method exploits the locality of hyperlinked pages. The source and destination pages for a hyperlink are often close to each other lexicographically. A page labeled 100 often has inlinks from pages nearby such as pages 112, 113, 116, and 117 rather than pages 117,924 and 4,931,010. Based on this locality principle, the information in an adjacency list for page 100 is stored as below.

Node	Inlinks from
100	112 0 2 0

Storing the gaps between pages compresses storage because these gaps are usually nice, small integers.

The reference encoding technique for graph compression exploits the similarity between webpages. If pages x and y have similar adjacency lists, it is possible to compress the adjacency list of y by representing it in terms of the adjacency list of x, in which case x is called a reference page for y. Pages within the same domain might often share common outlinks, making the reference encoding technique attractive. Consider the example in Figure 2, taken from [102]. The binary reference vector, which has the same size

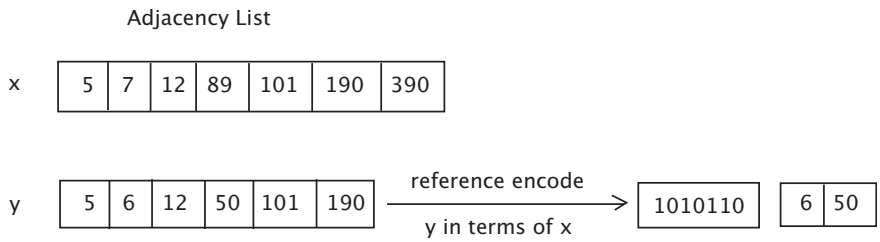

Figure 2: Reference encoding example

as the adjacency list of x, contains a 1 in the i^{th} position if the corresponding adjacency list entry i is shared by x and y. The second vector in the reference encoding is a list of all entries in the adjacency list of y that are not found in the adjacency list of its reference x. Reference encoding provides a nice means of compressing the data in an adjacency list, however, for each page one needs to determine which page should serve as the reference page. This is not an easy decision, but heuristics are suggested in [101]. Both the gap method and the reference encoding method are used, along with other compression techniques, to impressively compress the information in a standard web graph. These techniques are freely available in the graph compression tool WebGraph, which is produced by Paolo Boldi and Sebastiano Vigna [22, 23].

The final storage issue we discuss concerns dangling nodes. The pages of the web can be classified as either dangling nodes or nondangling nodes. Recall that dangling nodes are webpages that contain no outlinks. All other pages, having at least one outlink, are called nondangling nodes. Dangling nodes exist in many forms. For example, a page of data, a page with a postscript graph, a page with jpeg pictures, a pdf document, a page that has been fetched by a crawler but not yet explored–these are all examples of possible dangling nodes. As the research community moves more and more material online in the form of pdf and postscript files of preprints, talks, slides, and technical reports, the proportion of dangling nodes is growing. In fact, for some subsets of the web, dangling nodes make up 80% of the collection's pages.

The presence of these dangling nodes can cause philosophical, storage, and computational issues for the PageRank problem. We address the storage issue now and save the philosophical and computational issues associated with dangling nodes for the next section. Recall that Google founders Brin and Page suggested replacing $\mathbf{0}^T$ rows of the sparse hyperlink matrix $\mathbf{P}$ with dense vectors (the uniform vector $\frac{1}{n}\mathbf{e}^T$ or the more general $\mathbf{v}^T$ vector) to create the stochastic matrix $\bar{\mathbf{P}}$. Of course, if this suggestion were to be implemented explicitly, storage requirements would increase dramatically. Instead, the stochasticity fix can be modeled implicitly with the construction of one vector $\mathbf{a}$. Element $a_i = 1$ if row i of $\mathbf{P}$ corresponds to a dangling node, and 0, otherwise. Then $\bar{\mathbf{P}}$ (and also $\bar{\bar{\mathbf{P}}}$) can be written as a rank-one update of $\mathbf{P}$.

$$\bar{\mathbf{P}} = \mathbf{P} + \mathbf{a}\mathbf{v}^T, \text{ and therefore, } \bar{\bar{\mathbf{P}}} = \alpha\,\bar{\mathbf{P}} + (1-\alpha)\,\mathbf{e}\mathbf{v}^T = \alpha\,\mathbf{P} + (\alpha\,\mathbf{a} + (1-\alpha)\,\mathbf{e})\mathbf{v}^T.$$

5 Solution Methods for Solving the PageRank Problem

Regardless of the method for filling in and storing the entries of $\bar{\bar{\mathbf{P}}}$, PageRank is determined by computing the stationary solution $\boldsymbol{\pi}^T$ of the Markov chain. The row vector $\boldsymbol{\pi}^T$ can be found by solving either the eigenvector problem

$$\boldsymbol{\pi}^T\bar{\bar{\mathbf{P}}} = \boldsymbol{\pi}^T,$$

or by solving the homogeneous linear system

$$\boldsymbol{\pi}^T(\mathbf{I} - \bar{\bar{\mathbf{P}}}) = \mathbf{0}^T,$$

where $\mathbf{I}$ is the identity matrix. Both formulations are subject to an additional equation, the normalization equation $\boldsymbol{\pi}^T\mathbf{e} = 1$, where $\mathbf{e}$ is the column vector of all 1's. The normalization equation insures that $\boldsymbol{\pi}^T$ is a probability vector. The i^{th} element of $\boldsymbol{\pi}^T$, π_i, is the PageRank of page i. Stewart's book, "An Introduction to the Numerical Solution of Markov Chains" [108], contains an excellent presentation of the various methods of solving the Markov chain problem.

5.1 The Power Method

Traditionally, computing the PageRank vector has been viewed as an eigenvector problem, $\boldsymbol{\pi}^T\bar{\bar{\mathbf{P}}} = \boldsymbol{\pi}^T$, and the notoriously slow power method has been the method of choice. There are several good reasons for using the power method. First, consider iterates of the power method applied to $\bar{\bar{\mathbf{P}}}$ (a completely dense matrix, were it to be formed explicitly). Note that $\mathbf{E} = \mathbf{e}\mathbf{v}^T$. For any starting vector $\mathbf{x}^{(0)T}$ (generally, $\mathbf{x}^{(0)T} = \mathbf{e}^T/n$),

$$\begin{aligned}\mathbf{x}^{(k)T} &= \mathbf{x}^{(k-1)T}\bar{\bar{\mathbf{P}}} = \alpha\mathbf{x}^{(k-1)T}\bar{\mathbf{P}} + (1-\alpha)\mathbf{x}^{(k-1)T}\mathbf{e}\mathbf{v}^T \\ &= \alpha\mathbf{x}^{(k-1)T}\bar{\mathbf{P}} + (1-\alpha)\mathbf{v}^T \\ &= \alpha\mathbf{x}^{(k-1)T}\mathbf{P} + (\alpha\mathbf{x}^{(k-1)T}\mathbf{a} + (1-\alpha))\mathbf{v}^T, \qquad (1)\end{aligned}$$

since $\mathbf{x}^{(k-1)T}$ is a probability vector, and thus, $\mathbf{x}^{(k-1)T}\mathbf{e} = 1$. Written in this way, it becomes clear that the power method applied to $\bar{\bar{\mathbf{P}}}$ can be implemented with vector-matrix multiplications on the extremely sparse $\mathbf{P}$, and $\bar{\bar{\mathbf{P}}}$ and $\bar{\mathbf{P}}$ are never formed or stored. A matrix-free method such as the power method is required due to the size of the matrices and vectors involved (Google's index is currently 4.3 billion pages). Fortunately, since $\mathbf{P}$ is sparse, each vector-matrix multiplication required by the power method can be computed in $nnz(\mathbf{P})$ flops, where $nnz(\mathbf{P})$ is the number of nonzeros in $\mathbf{P}$. And since the average

number of nonzeros per row in $\mathbf{P}$ is 3-10, $O(nnz(\mathbf{P})) \approx O(n)$. Furthermore, at each iteration, the power method only requires the storage of one vector, the current iterate, whereas other accelerated matrix-free methods, such as restarted GMRES or BiCGStab, require storage of at least several vectors, depending on the size of the subspace chosen. Finally, the power method on Brin and Page's $\bar{\mathbf{P}}$ matrix converges quickly. Brin and Page report success using only 50 to 100 power iterations [27].

We return to the issue of dangling nodes now, this time discussing their philosophical complications. In one of their early papers [25], Brin and Page report that they "often remove dangling nodes during the computation of PageRank, then add them back in after the PageRanks have converged." From this vague statement it is hard to say exactly how Brin and Page were computing PageRank. But, we are certain that the removal of dangling nodes is not a fair procedure. Some dangling nodes should receive high PageRank. For example, a very authoritative pdf file could have many inlinks from respected sources, and thus, should receive a high PageRank. Simply removing the dangling nodes biases the PageRank vector unjustly. In fact, doing the opposite and incorporating dangling nodes adds little computational effort (see equation (1)), and further, can have a beneficial effect as it can lead to more efficient and accurate computation of PageRank. (See [79] and the next section.)

5.1.1 Check for Important Mathematical Properties associated with the Power Method

In this section, we check the mathematical properties of uniqueness, existence, and convergence to be sure that the PageRank power method of equation (1) will converge to the correct solution vector. The irreducibility of the matrix $\bar{\bar{\mathbf{P}}}$, compliments of the fudge factor matrix $\mathbf{E}$, guarantees the existence of the unique stationary distribution vector for the Markov equation. Convergence of the PageRank power method is governed by the primitivity of $\bar{\bar{\mathbf{P}}}$. Because the iteration matrix $\bar{\bar{\mathbf{P}}}$ is a stochastic matrix, the spectral radius $\rho(\bar{\bar{\mathbf{P}}})$ is 1. If this stochastic matrix is not primitive, it may have several eigenvalues on the unit circle, causing convergence problems for the power method. One such problem was identified by Brin and Page as a rank sink, a dangling node that keeps accumulating more and more PageRank at each iteration. This rank sink is actually an absorbing state of the Markov chain. More generally, a reducible matrix may contain an absorbing class that eventually sucks all the PageRank into states in its class. The web graph may contain several such classes and the long-run probabilities of the chain then depend greatly on the starting vector. Some states and classes may have 0 rank in the long-run, giving an undesirable solution and interpretation for the PageRank problem. However, the situation is much nicer and the convergence much cleaner for a primitive matrix.

A primitive stochastic matrix has only one eigenvalue on the unit circle, all other eigenvalues have modulus strictly less than one [89]. This means that the power method applied to a primitive stochastic matrix $\mathbf{P}$ is guaranteed to converge to the unique dominant eigenvector—the stationary vector $\boldsymbol{\pi}^T$ for the Markov matrix and the PageRank vector for the Google matrix. This is one reason why Brin and Page added the fudge factor matrix $\mathbf{E}$ forcing primitivity. As a result, there are no issues with convergence of the ranking vector, and any positive probability vector can be used to start the iterative process. A thorough paper by Farahat et al. [51] discusses uniqueness, existence, and convergence for several link analysis algorithms and their modifications, including PageRank and HITS.

Rate of Convergence

Even though the power method applied to the primitive stochastic matrix $\bar{\bar{\mathbf{P}}}$ converges to a unique PageRank vector, the rate of convergence is a crucial issue, especially considering the scope of the vector-matrix multiplications—it's on the order of billions since PageRank operates on Google's version of the full web. The asymptotic rate of convergence of the PageRank power method is governed by the subdominant eigenvalue of the transition matrix $\bar{\bar{\mathbf{P}}}$. Kamvar and Haveliwala [61] have proven that, regardless of the

value of the personalization vector $\mathbf{v}^T$ in $\mathbf{E} = \mathbf{e}\mathbf{v}^T$, this subdominant eigenvalue is equal to the scaling factor α for a reducible hyperlink matrix $\mathbf{P}$ and strictly less than α for an irreducible hyperlink matrix $\mathbf{P}$. Since the Web unaltered is reducible, we can conclude that the rate of convergence of the power method applied to $\bar{\bar{\mathbf{P}}}$ is the rate at which $\alpha^k \to 0$. This explains the reported quick convergence of the power method from section 5.1. Brin and Page, the founders of Google, use $\alpha = .85$. Thus, a rough estimate of the number of iterations needed to converge to a tolerance level $\boldsymbol{\tau}$ (measured by the residual, $\mathbf{x}^{(k)T}\bar{\bar{\mathbf{P}}} - \mathbf{x}^{(k)T} = \mathbf{x}^{(k+1)T} - \mathbf{x}^{(k)T}$) is $\frac{log_{10}\boldsymbol{\tau}}{log_{10}\alpha}$. For $\boldsymbol{\tau} = 10^{-6}$ and $\alpha = .85$, one can expect roughly $\frac{-6}{log_{10}.85} \approx 85$ iterations until convergence to the PageRank vector. For $\boldsymbol{\tau} = 10^{-8}$, about 114 iterations and for $\boldsymbol{\tau} = 10^{-10}$, about 142 iterations. Brin and Page report success using only 50 to 100 power iterations, implying that $\boldsymbol{\tau}$ could range from 10^{-3} to 10^{-7}.

This means Google can dictate the rate of convergence according to how small α is chosen to be. Consequently, Google engineers are forced to perform a delicate balancing act. The smaller α is, the faster the convergence, but the smaller α is, the less the true hyperlink structure of the web is used to determine webpage importance. And slightly different values for α can produce very different PageRanks. Moreover, as $\alpha \to 1$, not only does convergence slow drastically, but sensitivity issues begin to surface as well. (See sections 6.1 and 7.)

We now present a shorter alternative proof of the second eigenvalue of the PageRank matrix to that provided by Kamvar and Haveliwala [61]. Our proof also goes further and proves the relationship between the spectrum of $\bar{\mathbf{P}}$ and the spectrum of $\bar{\bar{\mathbf{P}}}$. To maintain generality, we use the generic personalization vector $\mathbf{v}^T$ rather than the uniform teleportation vector $\mathbf{e}^T/n$. The personalization vector is presented in detail in section 6.2.

Theorem 5.1 *Given the spectrum of the stochastic matrix $\bar{\mathbf{P}}$ is $\{1, \lambda_2, \lambda_3, \ldots, \lambda_n\}$, the spectrum of the primitive stochastic matrix $\bar{\bar{\mathbf{P}}} = \alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{e}\mathbf{v}^T$ is $\{1, \alpha\lambda_2, \alpha\lambda_3, \ldots, \alpha\lambda_n\}$, where $\mathbf{v}^T$ is a probability vector.*

Proof: Since $\bar{\mathbf{P}}$ is stochastic, $(1, \mathbf{e})$ is an eigenpair of $\bar{\mathbf{P}}$. Let $\mathbf{Q} = (\,\mathbf{e} \quad \mathbf{X}\,)$ be a nonsingular matrix which has the eigenvector $\mathbf{e}$ as its first column. Let $\mathbf{Q}^{-1} = \begin{pmatrix} \mathbf{y}^T \\ \mathbf{Y}^T \end{pmatrix}$. Then $\mathbf{Q}^{-1}\mathbf{Q} = \begin{pmatrix} \mathbf{y}^T\mathbf{e} & \mathbf{y}^T\mathbf{X} \\ \mathbf{Y}^T\mathbf{e} & \mathbf{Y}^T\mathbf{X} \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$ implies $\mathbf{y}^T\mathbf{e} = 1$ and $\mathbf{Y}^T\mathbf{e} = \mathbf{0}$, so that

$$\mathbf{Q}^{-1}\bar{\mathbf{P}}\mathbf{Q} = \begin{pmatrix} \mathbf{y}^T\mathbf{e} & \mathbf{y}^T\bar{\mathbf{P}}\mathbf{X} \\ \mathbf{Y}^T\mathbf{e} & \mathbf{Y}^T\bar{\mathbf{P}}\mathbf{X} \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{y}^T\bar{\mathbf{P}}\mathbf{X} \\ 0 & \mathbf{Y}^T\bar{\mathbf{P}}\mathbf{X} \end{pmatrix}.$$

Thus, $\mathbf{Y}^T\bar{\mathbf{P}}\mathbf{X}$ has eigenvalues $\lambda_2, \ldots, \lambda_n$. Applying the similarity transformation to $\bar{\bar{\mathbf{P}}} = \alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{e}\mathbf{v}^T$ gives

$$\begin{aligned}
\mathbf{Q}^{-1}(\alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{e}\mathbf{v}^T)\mathbf{Q} &= \alpha\mathbf{Q}^{-1}\bar{\mathbf{P}}\mathbf{Q} + (1-\alpha)\mathbf{Q}^{-1}\mathbf{e}\mathbf{v}^T\mathbf{Q} \\
&= \begin{pmatrix} \alpha & \alpha\mathbf{y}^T\bar{\mathbf{P}}\mathbf{X} \\ 0 & \alpha\mathbf{Y}^T\bar{\mathbf{P}}\mathbf{X} \end{pmatrix} + (1-\alpha)\begin{pmatrix} \mathbf{y}^T\mathbf{e} \\ \mathbf{Y}^T\mathbf{e} \end{pmatrix}(\,\mathbf{v}^T\mathbf{e} \quad \mathbf{v}^T\mathbf{X}\,) \\
&= \begin{pmatrix} \alpha & \alpha\mathbf{y}^T\bar{\mathbf{P}}\mathbf{X} \\ 0 & \alpha\mathbf{Y}^T\bar{\mathbf{P}}\mathbf{X} \end{pmatrix} + \begin{pmatrix} (1-\alpha) & (1-\alpha)\mathbf{v}^T\mathbf{X} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \\
&= \begin{pmatrix} 1 & \alpha\mathbf{y}^T\bar{\mathbf{P}}\mathbf{X} + (1-\alpha)\mathbf{v}^T\mathbf{X} \\ \mathbf{0} & \alpha\mathbf{Y}^T\bar{\mathbf{P}}\mathbf{X} \end{pmatrix}.
\end{aligned}$$

Therefore, the eigenvalues of $\bar{\bar{\mathbf{P}}} = \alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{e}\mathbf{v}^T$ are $\{1, \alpha\lambda_2, \alpha\lambda_3, \ldots, \alpha\lambda_n\}$. □

This theorem provides a more compact proof than that found in [61], showing that for a reducible $\bar{\mathbf{P}}$ with several unit eigenvalues, $\lambda_2(\bar{\bar{\mathbf{P}}}) = \alpha$.

Convergence Criteria

The power method applied to $\bar{\bar{\mathbf{P}}}$ is the predominant method for finding the important PageRank vector. Being an iterative method, the power method continues until some termination criterion is met. In a previous paragraph, we mentioned the traditional termination criterion for the power method: stop when the residual (as measured by the difference of successive iterates) is less than some predetermined tolerance. However, Haveliwala [58] has rightfully noted that the exact values of the PageRank vector are not as important as the correct ordering of the values in this vector. That is, iterate until the ordering of the approximate PageRank vector obtained by the power method converges. Considering the scope of the PageRank problem, saving just a handful of iterations is praiseworthy. Haveliwala's experiments show that the savings could be even more substantial on some datasets. As few as 10 iterations produced a good approximate ordering, competitive with the exact ordering produced by the traditional convergence measure. This raises several interesting issues: How does one measure the difference between two orderings? How does one determine when an ordering has converged satisfactorily? Several papers [44, 47, 48, 58, 60, 86] have provided a variety of answers to the question of comparing rank orderings, using such measures as Kendall's Tau, rank aggregation, and set overlap.

5.1.2 Acceleration Techniques for the PageRank Power Method

Despite the fact that the PageRank problem amounts to solving an old problem (computing the stationary vector of a Markov chain), the size of the matrix makes this old problem much more challenging. In fact, it has been dubbed "The World's Largest Matrix Computation" by Cleve Moler [92]. For this reason, some researchers have proposed quick approximations to the PageRank vector. Chris Ding and his coworkers [42, 43] suggested using a simple count of the number of inlinks to a webpage as an approximation to its PageRank. On their datasets, they found this very inexpensive measure approximated the exact PageRank well. However, a paper by Prabahkar Raghavan et al. disputes this claim noting that "there is very little correlation on the web graph between a node's in-degree and its PageRank" [96]. Intuitively, this makes sense. PageRank's thesis is that it is not the quantity of inlinks to a page that counts, but rather, the quality of inlinks.

While approximations to PageRank have not proved fruitful, other means of accelerating the computation of the exact rankings have. In fact, because the classical power method is known for its slow convergence, researchers immediately looked to other solution methods. However, the size and sparsity of the web matrix create limitations on the solution methods and have caused the predominance of the power method. This restriction to the power method has forced new research on the often criticized power method and has resulted in numerous improvements to the vanilla-flavored power method that are tailored to the PageRank problem. Since 1998, the resurgence in work on the power method has brought exciting, innovative twists to the old unadorned workhorse. As each iteration of the power method on a Web-sized matrix is so expensive, reducing the number of iterations by a handful can save hours of computation. Some of the most valuable contributions have come from researchers at Stanford who have discovered several methods for accelerating the power method. These acceleration methods can be divided into two classes: those that save time by reducing the work per iteration and those that aim to reduce the total number of iterations. These goals are often at odds with one another. For example, reducing the number of iterations usually comes at the expense of a slight increase in the work per iteration. As long as this overhead is minimal, the proposed acceleration is considered beneficial.

Reduction in work per iteration

Two methods have been proposed that clearly aim to reduce the work incurred at each iteration of the power method. The first method was proposed by Kamvar et al. [69] and is called adaptive PageRank. This method adaptively reduces the work at each iteration by taking a closer look at elements in the iteration vector. Kamvar et al. noticed that some pages converge to their PageRank values faster than other pages. As elements of the PageRank vector converge, the adaptive PageRank method "locks" them and does not use them in subsequent computations. This adaptive power method provides a small speedup in the computation of PageRank, by 17%. However, while this algorithm was shown to converge in practice on a handful of datasets, it was not proven to converge in theory.

The second acceleration method in this class was produced by another group at Stanford, this time led by Chris Lee. The algorithm of Lee et al. [79] partitions the Web into dangling and nondangling nodes and applies an aggregation method to this partition. Since Google's fix for dangling nodes produces a block of identical rows (a row of $\bar{\mathbf{P}}$ is $\mathbf{v}^T$ for each dangling node), a lumpable aggregation method can be solved exactly and efficiently. In effect, this algorithm reduces the large $n \times n$ problem to a much smaller $k \times k$ problem, where k is the number of non-dangling nodes on the Web. If $k = \frac{1}{s}n$, then the time until convergence is reduced approximately by a factor of s over the power method. In section 5.2, we describe a linear system formulation of Lee et al.'s Markov chain formulation of the lumpable PageRank algorithm.

Reduction in the number of iterations

In order to reduce the number of iterations required by the PageRank power method, Kamvar et al. [71] produced an extrapolation method derived from the classic Aitken's Δ^2 method. On the datasets tested, their extension to Aitken extrapolation, known as quadratic extrapolation, reduces PageRank computation time by 50-300% with minimal overhead.

The same group of Stanford researchers, Kamvar et al. [70] has produced one more contribution to the acceleration of PageRank. This method straddles the classes above because it uses aggregation to reduce both the number of iterations and the work per iteration. This very promising method, called BlockRank, is an aggregation method that lumps sections of the Web by hosts. BlockRank involves three main steps that work within the natural structure of the web. First, local PageRanks for pages in a host are computed independently using the link structure of the host. As a result, local PageRank vectors, which are smaller than the global PageRank vector, exist for each host. In the next step, these local PageRanks are weighted by the importance of the corresponding host. This host weight is found by forming a host aggregation matrix, the size of which is equal to the number of hosts. The stationary vector of the small host aggregation matrix gives the long-run proportion of time a random surfer spends on each host. Finally, the usual PageRank algorithm is run using the weighted aggregate of the local PageRank vectors as the starting vector. The BlockRank algorithm produced a speedup of a factor of 2 on some of their datasets. More recent, but very related, algorithms [29, 83] use similar aggregation techniques to exploit the Web's inherent power law structure to speed ranking computations.

Yet another group of researchers from Stanford, joined by IBM scientists, dropped the restriction to the power method. In their short paper, Arasu et al. [7] provide one small experiment with the Gauss-Seidel method applied to the PageRank problem. Bianchini et al. [19] suggest using the Jacobi method to compute the PageRank vector. Despite this progress, these are just beginnings. If the holy grail of real-time personalized search is ever to be realized, then drastic speed improvements must be made, perhaps by innovative new algorithms, or the simple combination of many of the current acceleration methods into one algorithm.

5.2 The Linear System Formulation

In 1998 Brin and Page posed the original formulation and subsequent solution of the PageRank problem in the Markov chain realm. Since then nearly all of the subsequent modifications and improvements to the solution method have remained in the Markov realm. Stepping outside, into the general linear system realm, presents interesting new research avenues and several advantages, which are described in this section.

We begin by formulating the PageRank problem as a linear system. The eigenvalue problem $\boldsymbol{\pi}^T(\alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{e}\mathbf{v}^T) = \boldsymbol{\pi}^T$ can be rewritten, with some algebra as,

$$\boldsymbol{\pi}^T(\mathbf{I} - \alpha\bar{\mathbf{P}}) = (1-\alpha)\mathbf{v}^T. \tag{2}$$

This system is always accompanied by the normalization equation $\boldsymbol{\pi}^T\mathbf{e} = 1$. Cleve Moler [93] and Bianchini et al. [19] appear to have been the first to suggest the linear system formulation in equation (2). We note some interesting properties of the coefficient matrix in this equation.

Properties of $(\mathbf{I} - \alpha\bar{\mathbf{P}})$:

1. $(\mathbf{I} - \alpha\bar{\mathbf{P}})$ is an **M**-matrix.[2]

   Proof: Straightforward from the definition of **M**-matrix given by Berman and Plemmons [14] or Meyer [89]. □

2. $(\mathbf{I} - \alpha\bar{\mathbf{P}})$ is nonsingular.

   Proof: See Berman and Plemmons [14] or Meyer [89]. □

3. The row sums of $(\mathbf{I} - \alpha\bar{\mathbf{P}})$ are $1 - \alpha$.

   Proof: $(\mathbf{I} - \alpha\bar{\mathbf{P}})\mathbf{e} = (1-\alpha)\mathbf{e}$. □

4. $\|\mathbf{I} - \alpha\bar{\mathbf{P}}\|_\infty = 1 + \alpha$, provided at least one nondangling node exists.

   Proof: The ∞-matrix norm is the maximum absolute row sum. If a page i has a positive number of outlinks, then the corresponding diagonal element of $\mathbf{I} - \alpha\bar{\mathbf{P}}$ is 1. All other off-diagonal elements are negative, but sum to α in absolute value. □

5. Since $(\mathbf{I} - \alpha\bar{\mathbf{P}})$ is an **M**-matrix, $(\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1} \geq 0$.

   Proof: Again, see Berman and Plemmons [14] or Meyer [89]. □

6. The row sums of $(\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1}$ are $\frac{1}{1-\alpha}$. Therefore, $\|(\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1}\|_\infty = \frac{1}{1-\alpha}$.

   Proof: This follows from points 3 and 5 above. □

7. Thus, the condition number[3] $\kappa_\infty(\mathbf{I} - \alpha\bar{\mathbf{P}}) = \frac{1+\alpha}{1-\alpha}$.

[2] Consider the real matrix $\mathbf{A}$ that has $a_{ij} \leq 0$ for all $i \neq j$ and $a_{ii} \geq 0$ for all i. $\mathbf{A}$ can be expressed as $\mathbf{A} = s\mathbf{I} - \mathbf{B}$, where $s > 0$ and $\mathbf{B} \geq 0$. When $s \geq \rho(\mathbf{B})$, the spectral radius of $\mathbf{B}$, $\mathbf{A}$ is called an **M**-matrix. **M**-matrices can be either nonsingular or singular.

[3] A nonsingular matrix $\mathbf{A}$ is ill-conditioned if a small relative change in $\mathbf{A}$ can produce a large relative change in $\mathbf{A}^{-1}$. The condition number of $\mathbf{A}$, given by $\kappa = \|\mathbf{A}\|\|\mathbf{A}^{-1}\|$, measures the degree of ill-conditioning. Condition numbers can be defined for each matrix norm [89].

Proof: By virtue of points 4 and 6 above, the condition number $\kappa_\infty(\mathbf{I}-\alpha\bar{\mathbf{P}}) = \|(\mathbf{I}-\alpha\bar{\mathbf{P}})\|_\infty \|(\mathbf{I}-\alpha\bar{\mathbf{P}})^{-1}\|_\infty = \frac{1+\alpha}{1-\alpha}$. □

These nice properties of $(\mathbf{I}-\alpha\bar{\mathbf{P}})$ cause us to wonder if similar properties hold for $(\mathbf{I}-\alpha\mathbf{P})$. Again, we return to the dangling nodes and their rank-one fix $\mathbf{a}\mathbf{v}^T$. Since $\bar{\mathbf{P}} = \mathbf{P} + \mathbf{a}\mathbf{v}^T$, $(\mathbf{I}-\alpha\bar{\mathbf{P}})$ is very dense if the number of dangling nodes, $nnz(\mathbf{a})$, is large. Using the rank-one dangling node trick, we can once again write the Pagerank problem in terms of the very sparse $\mathbf{P}$. The linear system of equation (2) can be rewritten as

$$\boldsymbol{\pi}^T(\mathbf{I}-\alpha\mathbf{P}-\alpha\mathbf{a}\mathbf{v}^T) = (1-\alpha)\mathbf{v}^T.$$

If we let $\boldsymbol{\pi}^T\mathbf{a} = \gamma$, then the linear system becomes

$$\boldsymbol{\pi}^T(\mathbf{I}-\alpha\mathbf{P}) = (1-\alpha+\alpha\gamma)\mathbf{v}^T.$$

The scalar γ holds the sum of the π_i for i in the set of dangling nodes. Since the normalization equation $\boldsymbol{\pi}^T\mathbf{e} = 1$ will be applied at the end, we can arbitrarily choose a convenient value for γ, say $\gamma = 1$. Thus, the sparse linear system formulation of the PageRank problem becomes

$$\boldsymbol{\pi}^T(\mathbf{I}-\alpha\mathbf{P}) = \mathbf{v}^T \quad \text{with} \quad \boldsymbol{\pi}^T\mathbf{e} = 1. \tag{3}$$

In addition, $(\mathbf{I}-\alpha\mathbf{P})$ has many of the same properties as $(\mathbf{I}-\alpha\bar{\mathbf{P}})$.

Properties of $(\mathbf{I}-\alpha\mathbf{P})$:

1. $(\mathbf{I}-\alpha\mathbf{P})$ is an **M**-matrix.
2. $(\mathbf{I}-\alpha\mathbf{P})$ is nonsingular.
3. The row sums of $(\mathbf{I}-\alpha\mathbf{P})$ are either $1-\alpha$ for nondangling nodes or 1 for dangling nodes.
4. $\|\mathbf{I}-\alpha\mathbf{P}\|_\infty = 1+\alpha$, provided $\mathbf{P}$ is nonzero.
5. Since $(\mathbf{I}-\alpha\mathbf{P})$ is an **M**-matrix, $(\mathbf{I}-\alpha\mathbf{P})^{-1} \geq 0$.
6. The row sums of $(\mathbf{I}-\alpha\mathbf{P})^{-1}$ are equal to 1 for the dangling nodes and less than or equal to $\frac{1}{1-\alpha}$ for the nondangling nodes.
7. The condition number $\kappa_\infty(\mathbf{I}-\alpha\mathbf{P}) \leq \frac{1+\alpha}{1-\alpha}$.
8. The row of $(\mathbf{I}-\alpha\mathbf{P})^{-1}$ corresponding to dangling node i is $\mathbf{e}_i^T$, where $\mathbf{e}_i$ is the i^{th} column of the identity matrix.

The last property of $(\mathbf{I}-\alpha\mathbf{P})^{-1}$ does not apply to $(\mathbf{I}-\alpha\bar{\mathbf{P}})^{-1}$. This additional property makes the computation of the PageRank vector especially efficient. Suppose the rows and columns of $\mathbf{P}$ are permuted (i.e., the indices are reordered) so that the rows corresponding to dangling nodes are at the bottom of the matrix.

$$\mathbf{P} = \begin{matrix} \\ \text{ND} \\ \text{D} \end{matrix} \begin{matrix} \begin{matrix} \text{ND} & \text{D} \end{matrix} \\ \begin{pmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \end{matrix},$$

where ND is the set of nondangling nodes and D is the set of dangling nodes. Then the coefficient matrix in the sparse linear system formulation becomes

$$(\mathbf{I} - \alpha\mathbf{P}) = \begin{pmatrix} \mathbf{I} - \alpha\mathbf{P}_{11} & -\alpha\mathbf{P}_{12} \\ \mathbf{0} & \mathbf{I} \end{pmatrix},$$

and the inverse of this matrix is

$$(\mathbf{I} - \alpha\mathbf{P})^{-1} = \begin{pmatrix} (\mathbf{I} - \alpha\mathbf{P}_{11})^{-1} & \alpha(\mathbf{I} - \alpha\mathbf{P}_{11})^{-1}\mathbf{P}_{12} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}.$$

Therefore, the PageRank vector $\boldsymbol{\pi}^T = \mathbf{v}^T(\mathbf{I} - \alpha\mathbf{P})^{-1}$ can be written as

$$\boldsymbol{\pi}^T = \left(\mathbf{v}_1^T(\mathbf{I} - \alpha\mathbf{P}_{11})^{-1} \quad \middle| \quad \alpha\mathbf{v}_1^T(\mathbf{I} - \alpha\mathbf{P}_{11})^{-1}\mathbf{P}_{12} + \mathbf{v}_2^T \right),$$

where the personalization vector $\mathbf{v}^T$ has been partitioned into nondangling ($\mathbf{v}_1^T$) and dangling ($\mathbf{v}_2^T$) sections. Note that $\mathbf{I} - \alpha\mathbf{P}_{11}$ inherits many of the properties of $\mathbf{I} - \alpha\mathbf{P}$, most especially nonsingularity. In summary, we now have an algorithm that computes the PageRank vector using only the nondangling portion of the web, exploiting the rank-one structure of the dangling node fix.

Algorithm 1:

1. Solve for $\boldsymbol{\pi}_1^T$ in $\boldsymbol{\pi}_1^T(\mathbf{I} - \alpha\mathbf{P}_{11}) = \mathbf{v}_1^T$.
2. Compute $\boldsymbol{\pi}_2^T = \alpha\boldsymbol{\pi}_1^T\mathbf{P}_{12} + \mathbf{v}_2^T$.
3. Normalize $\boldsymbol{\pi}^T = [\boldsymbol{\pi}_1^T \;\; \boldsymbol{\pi}_2^T]/\|[\boldsymbol{\pi}_1^T \;\; \boldsymbol{\pi}_2^T]\|_1$.

This algorithm is much simpler and cleaner, but equivalent, to the specialized iterative method proposed by Lee et al. [79] (and mentioned in section 5.1.2), which exploits the dangling nodes to reduce computation of the PageRank vector, sometimes by a factor of 1/5.

In [75], we propose that this process of locating zero rows be repeated recursively on smaller and smaller submatrices of $\mathbf{P}$, continuing until a submatrix is created that has no zero rows. The result of this process is a decomposition of the $\mathbf{P}$ matrix that looks like Figure 3. In fact, this process amounts to a simple reordering of the indices of the Markov chain. The left pane shows the original $\mathbf{P}$ matrix and the right pane is the reordered matrix according to the recursive dangling node idea. The dataset `California.dat` (available from `http://www.cs.cornell.edu/Courses/cs685/2002fa/`) is a typical subset of the web. It contains 9,664 nodes and 16,773 links, pertaining to the query topic of "california".

In general, after this symmetric reordering, the coefficient matrix of the linear system formulation of the PageRank problem (Equation (3)) has the following structure.

$$(\mathbf{I} - \alpha\mathbf{P}) = \begin{pmatrix} \mathbf{I} - \alpha\mathbf{P}_{11} & -\alpha\mathbf{P}_{12} & -\alpha\mathbf{P}_{13} & \cdots & -\alpha\mathbf{P}_{1b} \\ & \mathbf{I} & -\alpha\mathbf{P}_{23} & \cdots & -\alpha\mathbf{P}_{2b} \\ & & \mathbf{I} & \cdots & -\alpha\mathbf{P}_{3b} \\ & & & \ddots & \\ & & & & \mathbf{I} \end{pmatrix},$$

where b is the number of square diagonal blocks in the reordered matrix. Thus, the system in equation (3) after reordering can be solved by forward substitution. The only system that must be solved directly is the first subsystem, $\boldsymbol{\pi}_1^T(\mathbf{I} - \alpha\mathbf{P}_{11}) = \mathbf{v}_1^T$, where $\boldsymbol{\pi}^T$ and $\mathbf{v}^T$ have also been partitioned accordingly. The remaining subvectors of $\boldsymbol{\pi}^T$ are computed quickly and efficiently by forward substitution. In the

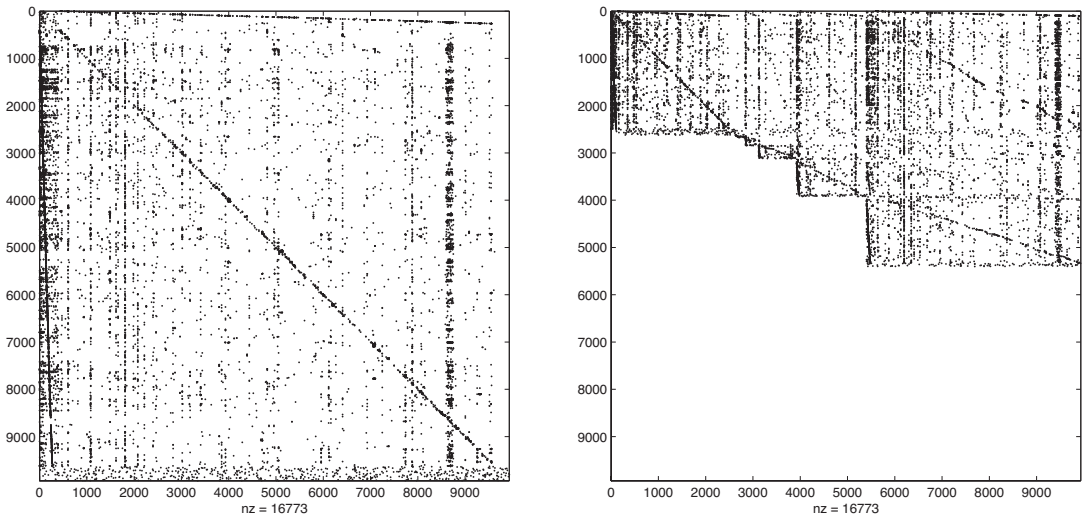

Figure 3: Original and reordered $\mathbf{P}$ matrix for `California.dat`

`California.dat` example, a $2,622 \times 2,622$ system can be solved instead of the full $9,664 \times 9,664$ system, or even the once-reduced $5,132 \times 5,132$ system. Using a direct method on the reordered linear system exploits dangling nodes, and is an extension to the dangling node power method suggested by Lee et al. [79]. The technical report [75] provides further details of the reordering method along with experimental results, suggested methods for solving the $\boldsymbol{\pi}_1^T(\mathbf{I} - \alpha\mathbf{P}_{11}) = \mathbf{v}_1^T$ system, and convergence properties.

In summary, this section and its linear system formulation open the door for many alternative solution methods, such as the iterative splittings of Jacobi and SOR, or even direct methods when the size of $\mathbf{P}_{11}$ is small enough. We expect that much more progress on the PageRank problem may be made now that researchers are no longer restricted to Markov chain methods and the power method.

6 Tinkering with the basic PageRank model

Varying α, although perhaps the most obvious alteration, is just one way to fiddle with the basic PageRank model presented in section 3. In this section, we explore several others, devoting a subsection to each.

6.1 Changing α

One of the most obvious places to begin fiddling with the basic PageRank model is α. Brin and Page, the founders of Google, have reported using $\alpha = .85$. One wonders why this choice for α? Might a different choice produce a very different ranking of retrieved webpages?

As mentioned in sections 5.1 and 5.1.1, there are good reasons for using $\alpha = .85$, one being the speedy convergence of the power method. With this value for α, we can expect the power method to converge to the PageRank vector in about 114 iterations for a convergence tolerance level of $\tau = 10^{-8}$. Obviously, this choice of α brings faster convergence than higher values of α. Compare with $\alpha = .99$, whereby roughly 1833 iterations are required to achieve a residual less than 10^{-8}. When working with

a sparse 4.3 billion by 4.3 billion matrix, each iteration counts; over a few hundred power iterations is more than Google is willing to compute. However, in addition to the computational reasons for choosing $\alpha = .85$, this choice for α also carries some intuitive weight: $\alpha = .85$ implies that roughly 5/6 of the time a Web surfer randomly clicks on hyperlinks (i.e., following the structure of the Web, as captured by the $\alpha\bar{\mathbf{P}}$ part of the formula), while 1/6 of the time this Web surfer will go to the URL line and type the address of a new page to "teleport" to (as captured by the $(1-\alpha)\mathbf{e}\mathbf{v}^T$ part of the formula). Perhaps this was the original motivation behind Brin and Page's choice of $\alpha = .85$; it produces an accurate model for Web surfing behavior. Whereas $\alpha = .99$, not only slows convergence of the power method, but also places much greater emphasis on the hyperlink structure of the Web and much less on the teleportation tendencies of surfers.

The PageRank vector derived from $\alpha = .99$ can be vastly different from that obtained using $\alpha = .85$. Perhaps it gives a "truer" PageRanking. Experiments with various α's show significant variation in rankings produced by different values of α [97, 98, 110]. As expected, the top section of the ranking changes only slightly, yet as we proceed down the ranked list we see more and more variation. Recall that the PageRank algorithm pulls a subset of elements from this ranked list, namely those elements that use or are related to the query terms. This means that the greater variation witnessed toward the latter half of the PageRank vector could lead to substantial variation in the ranking results returned to the user [97, 98]. Which ranking (i.e., which α) is preferred? This is a hard question to answer without doing extensive user verification tests on various datasets and queries. However, there are other ways to answer this question. In terms of convergence time, we've already emphasized the fact that $\alpha = .85$ is preferable, but later, in section 7, we present another good reason for choosing α near .85.

6.2 The personalization vector $\mathbf{v}^T$

One of the first modifications to the basic PageRank model suggested by its founders was a change to the teleportation matrix $\mathbf{E}$. Rather than using $\frac{1}{n}\mathbf{e}\mathbf{e}^T$, they used $\mathbf{e}\mathbf{v}^T$, where $\mathbf{v}^T > 0$ is a probability vector called the personalization or teleportation vector. Since $\mathbf{v}^T$ is a probability vector with positive elements, every node is still directly connected to every other node, thus, $\bar{\bar{\mathbf{P}}}$ is irreducible. Using $\mathbf{v}^T$ in place of $\frac{1}{n}\mathbf{e}^T$ means that the teleportation probabilities are no longer uniformly distributed. Instead, each time a surfer teleports, he or she follows the probability distribution given in $\mathbf{v}^T$ to jump to the next page. As shown in section 5.1, this slight modification retains the advantageous properties of the power method applied to $\bar{\bar{\mathbf{P}}}$. To produce a PageRank that is personalized for a particular user, only the constant vector $\mathbf{v}^T$ added at each iteration must be modified. (See equation (1).) Similarly, for the linear system formulation of the PageRank problem only the righthand side of the system changes for various personalized vectors $\mathbf{v}^T$.

It appears that the name personalization vector comes from the fact that Google intended to have many different $\mathbf{v}^T$ vectors for the many different classes of surfers. Surfers in one class, if teleporting, may be much more likely to jump to pages about sports, while surfers in another class may be much more likely to jump to pages pertaining to news and current events. Such differing teleportation tendencies can be captured in two different personalization vectors. This seems to have been Google's original intent in introducing the personalization vector [25]. However, it makes the once query-independent, user-independent PageRankings user-dependent and more calculation-laden. Nevertheless, it seems this little personalization vector has had more significant side effects. Google has recently used this personalization vector to control spamming done by the so-called link farms.

Link farms are set up by spammers to fool information retrieval systems into increasing the rank of their clients' pages. For example, suppose a business owner has decided to move a portion of his business online. The owner creates a webpage. However, this page rarely gets hits or is returned on web searches

on his product. So the owner contacts a search engine optimization company whose sole efforts are aimed at increasing the PageRank (and ranking among other search engines) of its clients' pages. One way a search engine optimizer attempts to do this is with link farms. Knowing that PageRank increases when the number of important inlinks to a client's page increases, optimizers add such links to a client's page. A link farm might have several interconnected nodes about important topics and with significant PageRanks. These interconnected nodes then link to a client's page, thus, in essence, sharing some of their PageRank with the client's page. The papers by Bianchini et al. [18, 19] present other scenarios for successfully boosting one's PageRank and provide helpful pictorial representations. Obviously, link farms are very troublesome for search engines. It appears that Google has tinkered with elements of $\mathbf{v}^T$ to annihilate the PageRank of link farms and their clients. Interestingly, this caused a court case between Google and the search engine optimization company SearchKing. The case ended in Google's favor [112].

Several researchers have taken the personalization idea beyond its spam prevention abilities, creating personalized PageRanking systems. Personalization is a hot area since some predict personalized engines as the future of search. See the Stanford research papers [41, 60, 62, 67, 103]. While the concept of personalization (producing a $\boldsymbol{\pi}^T$ for each user's $\mathbf{v}^T$ vector) sounds wonderful in theory, doing this in practice is computationally impossible. (Recall that it takes Google days to compute just one $\boldsymbol{\pi}^T$ corresponding to one $\mathbf{v}^T$ vector.) We focus on two papers that brings us closer to achieving the lofty goal of real-time personalized search engines. In [67], Jeh and Widom present their scalable personalized PageRank method. They identify a linear relationship between personalization vectors and their corresponding personalized PageRank vectors. This relationship allows the personalized PageRank vector to be expressed as a linear combination of vectors that Jeh and Widom call basis vectors. The number of basis vectors is a parameter in the algorithm. The computation of the basis vectors is reduced by the scalable dynamic programming approach described in [67]. At query time, an approximation to the personalized PageRank vector is constructed from the precomputed basis vectors. Their experiments show the promise of their approximations.

The second promising approach to achieving real-time personalized PageRank vectors can be found in [70]. The BlockRank algorithm of Kamvar et al. described in section 5.1.2 was originally designed as a method for accelerating the computation of the standard PageRank vector by finding a good starting vector, which it does quite well. However, one exciting additional consequence of this BlockRank algorithm is its potential use for personalization. BlockRank is an aggregation method that lumps sections of the Web by hosts, using the natural structure of the web. BlockRank involves three main steps. First, local PageRanks for pages in a host are computed independently using the link structure of the host. As a result, local PageRank vectors, which are smaller than the global PageRank vector, exist for each host. In the next step, these local PageRanks are weighted by the importance of the corresponding host. This host weight is found by forming an aggregation matrix, the size of which is equal to the number of hosts. Finally, the usual PageRank algorithm is run using the weighted aggregate of the local PageRank vectors as the starting vector. By assuming a web surfer can only teleport to hosts (rather than individual pages), personalization can be accounted for in the second step, in the formation of the aggregation matrix. The local PageRank vectors formed in the first step do not change, regardless of the host personalization vector. The final step of the personalized BlockRank algorithm proceeds as usual. This personalized BlockRank algorithm gives the personalized PageRank vector, not an approximation, with minimal overhead. However, while it does reduce the effort associated with personalized PageRank, it is still far from producing real-time personalized rankings.

We also note that as originally conceived, the PageRank model does not factor a web browser's back button into a surfer's hyperlinking possibilities. However, one team of researchers has made some theoretical progress on the insertion of the back button to the Markov model [46]. Several recent papers implement the back button in practical algorithms. One by Sydow [109] shows that an alternative ranking is provided by this adjustment, which appears to have a few advantages over the standard PageRanking. Another by Mathieu and Bouklit [84] uses a limited browser history stack to model a Markov chain with

finite memory.

6.3 Forcing Irreducibility

In the presentation of the PageRank model, we described the problem of reducibility. Simply put, the Markov chain produced from the hyperlink structure of the Web will almost certainly be reducible and thus a positive long-run stationary vector will not exist for the subsequent Markov chain. The original solution of Brin and Page uses the method of maximal irreducibility, whereby every node is directly connected to every other node, hence irreducibility is trivially enforced. However, maximal irreducibility does alter the true nature of the Web, whereas other methods of forcing irreducibility seem less invasive and more inline with the Web's true nature. We describe these alternative methods in turn, showing that they are equivalent, or nearly so, to Google's method of maximal irreducibility.

We refer to the first alternative as the method of minimal irreducibility [111]. In this method, a dummy node is added to the Web, which connects to every other node and to which every other node is connected, making the chain irreducible in a minimal sense. One way of creating a minimally irreducible $(n+1) \times (n+1)$ Markov matrix $\hat{\mathbf{P}}$ is

$$\hat{\mathbf{P}} = \left(\begin{array}{c|c} \alpha\bar{\mathbf{P}} & (1-\alpha)\mathbf{e} \\ \hline \mathbf{v}^T & 0 \end{array} \right).$$

This is clearly irreducible and primitive, and hence $\hat{\boldsymbol{\pi}}^T$, its corresponding PageRank vector, exists and can be found with the power method. State $n+1$ is a teleportation state. At any page, a random web surfer has a small probability $(1-\alpha)$ of transitioning to the teleportation state, from which point, he or she will teleport to one of the n original states according to the probabilities in the teleportation vector $\mathbf{v}^T$. We show that this minimally irreducible method is, in fact, equivalent to Google's maximally irreducible method. We examine the PageRank vector associated with this new $\hat{\mathbf{P}}$ (after the weight of $\hat{\pi}_{n+1}$, the PageRank of the dummy node, has been removed) as well as the convergence properties of the power method applied to $\hat{\mathbf{P}}$. We begin by comparing the spectrum of $\hat{\mathbf{P}}$ to the spectrum of $\bar{\mathbf{P}}$.

Theorem 6.1 *Given the stochastic matrix $\bar{\mathbf{P}}$ with spectrum $\{1, \lambda_2, \lambda_3, \ldots, \lambda_n\}$, the spectrum of $\hat{\mathbf{P}} = \left(\begin{array}{c|c} \alpha\bar{\mathbf{P}} & (1-\alpha)\mathbf{e} \\ \hline \mathbf{v}^T & 0 \end{array} \right)$ is $\{1, \alpha\lambda_2, \alpha\lambda_3, \ldots, \alpha\lambda_n, \alpha - 1\}$.*

Proof: Let $\mathbf{Q} = \begin{pmatrix} \mathbf{I} & \mathbf{e} \\ \mathbf{0}^T & 1 \end{pmatrix}$. Then $\mathbf{Q}^{-1} = \begin{pmatrix} \mathbf{I} & -\mathbf{e} \\ \mathbf{0}^T & 1 \end{pmatrix}$. The similarity transformation

$$\mathbf{Q}^{-1}\hat{\mathbf{P}}\mathbf{Q} = \begin{pmatrix} \alpha\bar{\mathbf{P}} - \mathbf{e}\mathbf{v}^T & \mathbf{0} \\ \mathbf{v}^T & 1 \end{pmatrix}.$$

Therefore, the spectrum $\sigma(\mathbf{Q}^{-1}\hat{\mathbf{P}}\mathbf{Q}) = \{1\} \cup \sigma(\alpha\bar{\mathbf{P}} - \mathbf{e}\mathbf{v}^T) = \{1, \alpha - 1, \alpha\lambda_2, \ldots, \alpha\lambda_n\}$. (The spectrum of $\alpha\bar{\mathbf{P}} - \mathbf{e}\mathbf{v}^T$ is $\{\alpha - 1, \alpha\lambda_2, \ldots, \alpha\lambda_n\}$ by the same trick used in the proof of Theorem 5.1.) □

Not only is the spectrum of the minimally irreducible $\hat{\mathbf{P}}$ nearly identical to the spectrum of the traditional $\bar{\mathbf{P}}$ used by Google, the PageRank vectors of the two systems are related.

Writing the power method on the partitioned matrix $\hat{\mathbf{P}}$ gives

$$(\hat{\boldsymbol{\pi}}^T \mid \hat{\pi}_{n+1}) = (\hat{\boldsymbol{\pi}}^T \mid \hat{\pi}_{n+1}) \left(\begin{array}{c|c} \alpha\bar{\mathbf{P}} & (1-\alpha)\mathbf{e} \\ \hline \mathbf{v}^T & 0 \end{array} \right),$$

which gives the following system of equations.

$$\hat{\boldsymbol{\pi}}^T = \alpha\hat{\boldsymbol{\pi}}^T\bar{\mathbf{P}} + \hat{\pi}_{n+1}\mathbf{v}^T, \tag{4}$$

$$\hat{\pi}_{n+1} = (1-\alpha)\hat{\boldsymbol{\pi}}^T\mathbf{e}. \tag{5}$$

Solving for $\hat{\pi}_{n+1}$ in equation (5) gives $\hat{\pi}_{n+1} = \frac{1-\alpha}{2-\alpha}$. Backsubstituting this value for $\hat{\pi}_{n+1}$ into equation (4) gives

$$\hat{\boldsymbol{\pi}}^T = \alpha\hat{\boldsymbol{\pi}}^T\bar{\mathbf{P}} + \frac{1-\alpha}{2-\alpha}\mathbf{v}^T. \tag{6}$$

Now the question is: how does $\hat{\boldsymbol{\pi}}^T$ relate to $\boldsymbol{\pi}^T$? Since state $n+1$ is an artificial state, we can remove its PageRank $\hat{\pi}_{n+1}$ and normalize the remaining subvector $\hat{\boldsymbol{\pi}}^T$. This means $\hat{\boldsymbol{\pi}}^T$ is multiplied by $\frac{1}{1-\hat{\pi}_{n+1}} = 2-\alpha$. Replacing $\hat{\boldsymbol{\pi}}^T$ in (6) with $(2-\alpha)\hat{\boldsymbol{\pi}}^T$ gives

$$\hat{\boldsymbol{\pi}}^T = \alpha\hat{\boldsymbol{\pi}}^T\bar{\mathbf{P}} + (1-\alpha)\mathbf{v}^T,$$

which is the exact formulation of the traditional maximally irreducible power method given in equation (1). Therefore, the particular method of minimal irreducibility turns out to be equivalent in theory and in computational efficiency to Google's method of maximal irreducibility. This is not surprising since intuitively both methods model teleportation in the same way.

There are other means of forcing irreducibility . However, some of these methods require classification and location of the states of the chain into essential and transient classes, and thus, can be more computationally intensive than the methods discussed in this section. Lastly, we mention an approach that, rather than forcing irreducibility on the web matrix, instead exploits the reducibility inherent in the web. Avrachenkov et al. [8] create a decomposition of the reducible matrix $\mathbf{P}$. The global PageRank solution can be found in a computationally efficient manner by computing the subPageRank of each connected component, then pasting the subPageRanks together to form the global PageRank. Identification of the connected components of the web graph can be determined by a graph traversal algorithm such as breadth-first search or depth-first search, which requires $O(n(\mathbf{P}) + nnz(\mathbf{P}))$ time. Then the computation of the subPageRank for each connected component can be done in parallel requiring $O(n(P_{CC}))$ time, where $n(P_{CC})$ is the size of the largest connected component. This is theoretically promising, however, the bowtie structure discovered by Broder et al. [28] shows that the largest connected component for a web graph is composed of nearly 30% of the nodes, so the savings are not overwhelming.

7 Sensitivity, Stability, and Condition Numbers

Section 6 discussed ideas for changing some parameters in the PageRank model. A natural question is how such changes affect the PageRank vector. Regarding the issues of sensitivity and stability, one would like to know how changes in $\mathbf{P}$ affect $\boldsymbol{\pi}^T$. The two different formulations of the PageRank problem, the linear system formulation and the eigenvector formulation, give some insight. The PageRank problem in its general linear system form is

$$\boldsymbol{\pi}^T(\mathbf{I} - \alpha\bar{\mathbf{P}}) = (1-\alpha)\mathbf{v}^T.$$

Section 5.2 listed a property pertaining to the condition number of the linear system, $\kappa_\infty(\mathbf{I}-\alpha\bar{\mathbf{P}}) = \frac{1+\alpha}{1-\alpha}$. (Also proven in [68].) As $\alpha \to 1$, the linear system becomes more ill-conditioned, meaning that a small change in the coefficient matrix creates a large change in the solution vector. However, $\boldsymbol{\pi}^T$ is actually an eigenvector for the corresponding Markov chain. While elements in the solution vector may change greatly for small changes in the coefficient matrix, the direction of the vector may change minutely. Once the solution is normalized to create a probability vector, the effect is minimal. The ill-conditioning of the linear system does not imply that the corresponding eigensystem is ill-conditioned, a fact documented by Wilkinson [115] (with respect to the inverse iteration method).

To answer the questions about how changes in $\mathbf{P}$ affect $\boldsymbol{\pi}^T$ what we need to examine is eigenvector sensitivity, not linear system sensitivity. A crude statement about eigenvector sensitivity is that if a simple eigenvalue is close to the other eigenvalues, then the corresponding eigenvector is sensitive to perturbations in $\mathbf{P}$, but a large gap does not insure insensitivity.

More rigorous measures of eigenvector sensitivity for Markov chains were developed by Meyer and Stewart [91], Meyer and Golub [57], Cho and Meyer [34], and Funderlic and Meyer [54]. While not true for general eigenanalysis, it is known [87] that for a Markov chain with matrix $\mathbf{P}$ the sensitivity of $\boldsymbol{\pi}^T$ to perturbations in $\mathbf{P}$ is governed by how close the subdominant eigenvalue λ_2 of $\mathbf{P}$ is to 1. Therefore, as α increases, the PageRank vector becomes more and more sensitive to small changes in $\mathbf{P}$. Thus, Google's choice of $\alpha = .85$, while staying further from the true hyperlink structure of the Web, gives a much more stable PageRank than the "truer to the Web" choice of $\alpha = .99$.

This same observation can be arrived at alternatively using derivatives. The parameter α is usually set to .85, but it can theoretically vary between $0 < \alpha < 1$. Of course, $\bar{\bar{\mathbf{P}}}$ depends on α, and so, $\bar{\bar{\mathbf{P}}}(\alpha) = \alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{e}\mathbf{v}^T$. The question about how sensitive $\boldsymbol{\pi}^T(\alpha)$ is to changes in α can be answered precisely if the derivative $d\boldsymbol{\pi}^T(\alpha)/d\alpha$ can be evaluated. But before attempting to differentiate we should be sure that this derivative is well-defined. The distribution $\boldsymbol{\pi}^T(\alpha)$ is a left-hand eigenvector for $\bar{\bar{\mathbf{P}}}(\alpha)$, but eigenvector components need not be differentiable (or even continuous) functions of the entries of $\bar{\bar{\mathbf{P}}}(\alpha)$ [89, pg. 497], so the existence of $d\boldsymbol{\pi}^T(\alpha)/d\alpha$ is not a slam dunk. The following theorem provides what is needed.

Theorem 7.1 *The PageRank vector is given by*

$$\boldsymbol{\pi}^T(\alpha) = \frac{1}{\sum_{i=1}^{n} D_i(\alpha)} \big(D_1(\alpha),\, D_2(\alpha),\, \ldots,\, D_n(\alpha)\big),$$

where $D_i(\alpha)$ is the i^{th} principal minor determinant of order $n-1$ in $\mathbf{I} - \bar{\bar{\mathbf{P}}}(\alpha)$. Because each principal minor $D_i(\alpha) > 0$ is just a sum of products of numbers from $\mathbf{I} - \bar{\bar{\mathbf{P}}}(\alpha)$, it follows that each component in $\boldsymbol{\pi}^T(\alpha)$ is a differentiable function of α on the interval $(0,1)$.

Proof: For convenience, let $\mathbf{P} = \bar{\bar{\mathbf{P}}}(\alpha)$, $\boldsymbol{\pi}^T(\alpha) = \boldsymbol{\pi}^T$, $D_i = D_i(\alpha)$, and set $\mathbf{A} = \mathbf{I} - \mathbf{P}$. If $\text{adj}\,(\mathbf{A})$ denotes the transpose of the matrix of cofactors (often called the adjugate or adjoint), then

$$\mathbf{A}[\text{adj}\,(\mathbf{A})] = \mathbf{0} = [\text{adj}\,(\mathbf{A})]\mathbf{A}.$$

It follows from the Perron–Frobenius theorem that $rank\,(\mathbf{A}) = n-1$, and hence $rank\,(\text{adj}\,(\mathbf{A})) = 1$. Furthermore, Perron–Frobenius insures that each column of $[\text{adj}\,(\mathbf{A})]$ is a multiple of $\mathbf{e}$, so $[\text{adj}\,(\mathbf{A})] = \mathbf{e}\mathbf{w}^T$ for some vector $\mathbf{w}$. But $[\text{adj}\,(\mathbf{A})]_{ii} = D_i$, so $\mathbf{w}^T = (D_1,\, D_2,\, \ldots,\, D_n)$. Similarly, $[\text{adj}\,(\mathbf{A})]\mathbf{A} = \mathbf{0}$ insures that each row in $[\text{adj}\,(\mathbf{A})]$ is a multiple of $\boldsymbol{\pi}^T$ and hence $\mathbf{w}^T = \alpha\boldsymbol{\pi}^T$ for some α. This scalar α can't be zero; otherwise $[\text{adj}\,(\mathbf{A})] = \mathbf{0}$, which is impossible. Therefore, $\mathbf{w}^T\mathbf{e} = \alpha \neq 0$, and $\mathbf{w}^T/(\mathbf{w}^T\mathbf{e}) = \mathbf{w}^T/\alpha = \boldsymbol{\pi}^T$. ∎ □

Theorem 7.2 *If* $\boldsymbol{\pi}^T(\alpha) = \big(\pi_1(\alpha), \pi_2(\alpha), \ldots \pi_n(\alpha)\big)$ *is the PageRank vector , then*

$$\left|\frac{d\pi_j(\alpha)}{d\alpha}\right| \leq \frac{1}{1-\alpha} \qquad \textit{for each } j = 1, 2, \ldots, n, \tag{7}$$

and

$$\left\|\frac{d\boldsymbol{\pi}^T(\alpha)}{d\alpha}\right\|_1 \leq \frac{2}{1-\alpha}. \tag{8}$$

Proof: First compute $d\boldsymbol{\pi}^T(\alpha)/d\alpha$ by noting that $\boldsymbol{\pi}^T(\alpha)\mathbf{e} = 1$ implies

$$\frac{d\boldsymbol{\pi}^T(\alpha)}{d\alpha}\mathbf{e} = 0.$$

Using this while differentiating both sides of

$$\boldsymbol{\pi}^T(\alpha) = \boldsymbol{\pi}^T(\alpha)\big(\alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{e}\mathbf{v}^T\big)$$

yields

$$\frac{d\boldsymbol{\pi}^T(\alpha)}{d\alpha}(\mathbf{I} - \alpha\bar{\mathbf{P}}) = \boldsymbol{\pi}^T(\alpha)(\bar{\mathbf{P}} - \mathbf{e}\mathbf{v}^T).$$

Matrix $\mathbf{I} - \alpha\bar{\mathbf{P}}(\alpha)$ is nonsingular because $\alpha < 1$ guarantees that $\rho\big(\alpha\bar{\mathbf{P}}(\alpha)\big) < 1$, so

$$\frac{d\boldsymbol{\pi}^T(\alpha)}{d\alpha} = \boldsymbol{\pi}^T(\alpha)(\bar{\mathbf{P}} - \mathbf{e}\mathbf{v}^T)(\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1}. \tag{9}$$

The proof of (7) hinges on the following inequality. For every real $\mathbf{x} \in \mathbf{e}^{\perp}$ (the orthogonal complement of $span\{\mathbf{e}\}$), and for all real vectors $\mathbf{y}_{n\times 1}$,

$$|\mathbf{x}^T\mathbf{y}| \leq \|\mathbf{x}\|_1 \left(\frac{y_{\max} - y_{\min}}{2}\right). \tag{10}$$

This is a consequence of Hölder's inequality because for all real α,

$$|\mathbf{x}^T\mathbf{y}| = |\mathbf{x}^T(\mathbf{y} - \alpha\mathbf{e})| \leq \|\mathbf{x}\|_1 \|\mathbf{y} - \alpha\mathbf{e}\|_\infty,$$

and $\min_\alpha \|\mathbf{y} - \alpha\mathbf{e}\|_\infty = (y_{\max} - y_{\min})/2$, where the minimum is attained at $\alpha = (y_{\max} + y_{\min})/2$. It follows from (9) that

$$\frac{d\pi_j(\alpha)}{d\alpha} = \boldsymbol{\pi}^T(\alpha)(\bar{\mathbf{P}} - \mathbf{e}\mathbf{v}^T)(\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1}\mathbf{e}_j,$$

where $\mathbf{e}_j$ is the j^{th} standard basis vector (i.e, the j^{th} column of $\mathbf{I}_{n\times n}$). Since $\boldsymbol{\pi}^T(\alpha)(\bar{\mathbf{P}} - \mathbf{e}\mathbf{v}^T)\mathbf{e} = 0$, inequality (10) may be applied with

$$\mathbf{y} = (\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1}\mathbf{e}_j$$

to obtain

$$\left|\frac{d\pi_j(\alpha)}{d\alpha}\right| \leq \|\boldsymbol{\pi}^T(\alpha)(\bar{\mathbf{P}} - \mathbf{e}\mathbf{v}^T)\|_1 \left(\frac{y_{\max} - y_{\min}}{2}\right).$$

But $\|\boldsymbol{\pi}^T(\alpha)(\bar{\mathbf{P}} - \mathbf{e}\mathbf{v}^T)\|_1 \leq 2$, so

$$\left|\frac{d\pi_j(\alpha)}{d\alpha}\right| \leq y_{\max} - y_{\min}.$$

Now use the fact that $(\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1} \geq \mathbf{0}$ together with the observation that

$$(\mathbf{I} - \alpha\bar{\mathbf{P}})\mathbf{e} = (1-\alpha)\mathbf{e} \implies (\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1}\mathbf{e} = (1-\alpha)^{-1}\mathbf{e}$$

to conclude that $y_{\min} \geq 0$ and

$$y_{\max} \leq \max_{i,j} \left[(\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1}\right]_{ij} \leq \|(\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1}\|_\infty = \|(\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1}\mathbf{e}\|_\infty = \frac{1}{1-\alpha}.$$

Consequently,

$$\left|\frac{d\pi_j(\alpha)}{d\alpha}\right| \leq \frac{1}{1-\alpha},$$

which is (7). Inequality (8) is a direct consequence of (9) along with the above observation that

$$\|(\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1}\|_\infty = \|(\mathbf{I} - \alpha\bar{\mathbf{P}})^{-1}\mathbf{e}\|_\infty = \frac{1}{1-\alpha}. \quad \blacksquare$$

□

Theorem 7.2 makes it apparent that the sensitivity of the PageRank vector as a function of α is primarily governed by the size of $(1-\alpha)^{-1}$. If is α close to 1, then PageRank is sensitive to small changes in α. Therefore, there is a balancing act to be performed. As α becomes smaller, the influence of the the actual link structure in the web is decreased and effects of the artificial probability $\mathbf{v}^T$ are increased. Since PageRank is trying to take advantage of the underlying link structure, it is more desirable (at least in this respect) to choose α close to 1. However, it α is too close to 1, then, as we have just observed, PageRanks will be unstable, and the convergence rate slows.

Three other research groups have examined the sensitivity and stability of the PageRank vector; Ng et al. at the University of California at Berkeley, Bianchini et al. in Siena, Italy and Borodin et al. at the University of Toronto. All three groups have computed bounds on the difference between the old PageRank vector $\boldsymbol{\pi}^T$ and the new, updated PageRank vector $\tilde{\boldsymbol{\pi}}^T$. Using Aldous' notion of variational distance [5], Ng et al. [94] arrive at

$$\|\boldsymbol{\pi}^T - \tilde{\boldsymbol{\pi}}^T\|_1 \leq \frac{2}{1-\alpha} \sum_{i \in U} \pi_i,$$

where U is the set of all pages that have been updated. Bianchini et al. [19], using concepts of energy flow, and Borodin et al. [80] improve upon this bound, replacing $\frac{2}{1-\alpha}$ with $\frac{2\alpha}{1-\alpha}$. The interpretation is that as long as α is not close to 1 and the updated pages do not have high PageRank, then the updated PageRanks do not change much. For $\alpha = .85$, $\frac{2\alpha}{1-\alpha} = 11.\overline{3}$, which means that the 1-norm of the difference between the old PageRank vector and the new, updated PageRank vector is less than $11.\overline{3}$ times the sum of the old PageRank for all updated pages. All three groups use the bounds to conclude that PageRank is "robust" and "stable", compared to other ranking algorithms such as HITS. However, being more stable than another algorithm only makes the algorithm in question comparatively stable not uniformly stable. In fact, Bianchini et al. [19] "highlight a nice property of PageRank, namely that a community can only make a very limited change to the overall PageRank of the Web. Thus, regardless of the way they change, non-authoritative communities cannot affect significantly the global PageRank." On the other hand, authoritative communities whose high ranking pages are updated can significantly affect the global PageRank. The experiments done by the Berkeley group involve removing a random 30% of their dataset and recomputing the importance vector [95]. (The Toronto group conducted similar experiments on much smaller datasets [80].) Their findings show that PageRank is stable under such perturbation. However, we contest that these results may be misleading. As stated aptly by the Italian researchers, perturbations

to non-authoritative pages have little effect on the rankings. Removing a random portion of the graph amounts to removing a very large proportion of non-authoritative pages compared to authoritative pages, due to the Web's scale-free structure [11]. (A more detailed description of the scale-free structure of the Web comes in section 9.) A better indication of PageRank's stability (or any ranking algorithm's stability) is its sensitivity to carefully selected perturbations, namely perturbations of the hubs or high PageRank pages. In fact, this paints a much more realistic picture as these are the most likely to change and most frequently changing pages on the Web [53].

A fourth group of researchers recently joined the stability discussion. Lempel and Moran, the inventors of the SALSA algorithm [81], have added a further distinction to the definition of stability. In [82], they note that stability of an algorithm, which concerns volatility of the *scores* assigned to pages, has been well-studied. What has not been studied is the notion of rank-stability (first defined and studied by Borodin et al. [24], which addresses how volatile the *rankings* of pages are with respect to changes in the underlying graph. Lempel and Moran show that stability does not imply rank-stability. In fact, they provide a small example demonstrating that a change in one outlink of a very low ranking page can turn the entire ranking upside down! They also introduce the interesting concept of running-time stability, challenging researchers to examine the effect of small perturbations in the graph on an algorithm's running time.

8 Updating the PageRank vector

Section 7 gave a brief introduction to the updating problem. Here we present a more thorough analysis. We begin by emphasizing the need for updating the PageRank vector frequently. A study by Cho and Garcia-Molina [36] in 2000 reported that 40% of all webpages in their dataset changed within a week, and 23% of the .com pages changed daily. In a much more extensive and recent study, the results of Fetterly et al. [53] concur. About 35% of all webpages changed over the course of their study, and also pages that were larger in size changed more often and more extensively than their smaller counterparts. In the above studies, change was defined as either a change in page content or a change in page outlinks or both. Now consider news webpages, where updates to both content and links might occur on an hourly basis. Clearly, the PageRank vector must be as dynamic as the Web. Currently, Google updates its PageRank vector monthly [2]. Researchers have been working to make updating easier, taking advantage of old computations to speed updated computations. To our knowledge, the PageRank vector for the entire web is recomputed each month from scratch. (Popular sites may have their PageRank updated more frequently.) That is, last month's vector is not used to create this month's vector. A Google spokesperson at the annual SIAM meeting in 2002 reported that restarting this month's power method with last month's vector seemed to provide no improvement. This implies that the two vectors are just not close enough to each other for the restarted power method to effect any gains.

In general, the updating problem is stated as: given an old Markov matrix $\mathbf{P}$ and its stationary vector $\boldsymbol{\pi}^T$ along with the updated Markov matrix $\tilde{\mathbf{P}}$, find the updated stationary vector $\tilde{\boldsymbol{\pi}}^T$. There are several updating methods for finding $\tilde{\boldsymbol{\pi}}^T$ when the updates affect only elements of $\mathbf{P}$ (as opposed to the addition or deletion of states, which change the size of $\mathbf{P}$). The simplest updating approach begins an iterative method applied to $\tilde{\mathbf{P}}$ with $\boldsymbol{\pi}^T$ as the starting vector. Intuition counsels that if $\tilde{\mathbf{P}} \approx \mathbf{P}$, then $\tilde{\boldsymbol{\pi}}^T$ should be close to $\boldsymbol{\pi}$ and can thus be obtained, starting from $\boldsymbol{\pi}^T$, with only a few more iterations of the chosen iterative method. However, unless $\boldsymbol{\pi}^T$ is very close to $\tilde{\boldsymbol{\pi}}^T$, this takes as many iterations as using a random or uniform starting vector. Apparently, this is what PageRank engineers have witnessed. Since the PageRank updating problem is really a Markov chain with a particular form, we begin by reviewing Markov chain updating techniques. Markov chain researchers have been studying the updating problem for some time, hoping to find $\tilde{\boldsymbol{\pi}}^T$ inexpensively without resorting to full recomputation. There have been many papers on the topic of perturbation bounds for the stationary solution of a Markov chain

[35, 54, 57, 66, 88, 106]. These bounds are similar in spirit to the bounds of Ng et al. [94] and Bianchini et al. [19] presented in section 7. These papers aim to produce tight bounds on the difference between $\boldsymbol{\pi}^T$ and $\tilde{\boldsymbol{\pi}}^T$, showing that the magnitude of the changes in $\mathbf{P}$ gives information about the sensitivity of elements of $\boldsymbol{\pi}^T$. However, there are some papers whose aim is to produce more than just bounds; these papers show exactly how changes in $\mathbf{P}$ affect each element in $\boldsymbol{\pi}^T$. One expensive method uses the group inverse to update the static chain [90]. Calculating the group inverse for a web-sized matrix is not a practical option. Similar analyses use mean first passage times, the fundamental matrix, or an $\mathbf{LU}$ factorization to update $\boldsymbol{\pi}^T$ exactly [35, 55, 73, 106]. Yet these are also expensive means of obtaining $\tilde{\boldsymbol{\pi}}^T$ and remain computationally impractical. These classical Markov chain updating methods are also considered static, in that they only accommodate updates to the elements of the matrix, state additions and deletions cannot be handled. Thus, due to the dynamics of the Web, these computationally impractical methods also have theoretical limitations. New updating methods that handle dynamic Markov chains must be developed.

The first updating paper [33] aimed specifically at the PageRank problem and its dynamics was available online in early 2002 and was the work of Steve Chien, a Berkeley student, Cynthia Dwork from Microsoft, and Kumar and Sivakumar of IBM Almaden. These researchers created an algorithm that provided a fast approximate PageRank for updates to the Web's link structure. The intuition behind their algorithm was the following: identify a small portion of the Web graph "near" the link changes and model the rest of the Web as a single node in a new, much smaller graph; compute a PageRank vector for this small graph and transfer these results to the much bigger, original graph. Their results, although only handling link updates, not state updates, were quite promising. So much so, that we recognized the potential for improvement to their algorithm. In [76, 77], we outlined the connection between the algorithm of Chien et al. and aggregation methods. In fact, Chien et al. essentially complete one step of an aggregation method. We formalized the connection and produced a specialized iterative aggregation algorithm for updating any Markov chain with any type of update, link or state. This iterative aggregation algorithm works especially well on the PageRank problem due to the graph's underlying scale-free structure. (More on the scale-free properties can be found in section 9.) Our updating algorithm produced speedups on the order of 5-10. Even greater potential for speedup exists since the other power method acceleration methods of section 5.1.2 can be used in conjunction with our method. While our updating solution can be applied to any Markov chain, other updating techniques tailored completely to the PageRank problem exist [3, 19, 70, 113]. These techniques often use the crawlers employed by the search engine to adaptively update PageRank approximately, without requiring storage of the transition matrix. Although the dynamic nature of the Web creates challenges, it has pushed researchers to develop better solutions to the old problem of updating the stationary vector of a Markov chain. Other areas for improvement are detailed in the next section.

9 Areas of Future Research

9.1 Storage and Speed

Two areas of current research, storage and computational speed, will remain areas of future work for some time. As the Web continues its amazing growth, the need for smarter storage schemes and even faster numerical methods will become more evident. Both are exciting areas for computer scientists and numerical analysts interested in information retrieval.

9.2 Spam

Another area drawing attention recently is spam identification and prevention. This was cited by Monika Henzinger, former Research Director at Google, as a present "challenge" in an October 2002 paper [65]. Once thought to be impervious to spamming, researchers have been revealing subtle ways of boosting PageRank [19, 113]. The paper by Bianchini et al. [19], based on its suggested ways to alter PageRank, goes on to describe how to identify spamming techniques, such as link farms, which can take the form of a regular graph. This is a first step toward preventing spam. However, as long as the Web provides some mercantile potential, search engine optimization companies will exist and the papers they write for spammers will circulate. At least a dozen or so papers with nearly the same title exist for spammers, "PageRank Explained and How to Make the Most of it" [1, 39, 104, 105]. Clearly, this makes for an ongoing war between search engines and the optimization companies and requires constant tweaking of the underlying algorithms in an attempt to outwit the spammers.

9.3 The evolution and dynamics of the Web

Viewing the Web as a dynamic organism introduces some interesting areas of research. The Web's constant growth and frequent updates create an *evolving network*, as opposed to a static network. Adaptive algorithms have been presented to accommodate for this evolution [3, 53, 113]. Google itself has begun research on "stream of text" information such as news and TV broadcasts. Such dynamic content creates challenges that need tailored solutions. One example is the query-free news search proposed by Google engineers in [64]. This is related to the algorithmic challenge of using changes in data streams to locate interesting trends, a challenge identified by Monika Henzinger in her 2003 paper, "Algorithmic Challenges in Web Search Engines" [63].

9.4 Structure on many levels

A final prediction for future research is the exploitation of the Web's structure in all aspects of information retrieval. The Web has structure on many different levels. A level discovered in 2000 by Broder et al. [28] and often cited since is the *bowtie* structure of the Web. Their findings show that nearly a quarter of the Web is composed of one giant strongly connected component, one-fifth is composed of pages pointing into the strongly connected component, another one-fifth of pages pointing out from the strongly connected component, another one-fifth is composed of pages called tendrils and the remaining Web consists of disconnected pages. Arasu et al. [7] propose an algorithm that computes PageRank more efficiently by exploiting this bowtie structure. Dill et al. discovered that the bowtie structure is self-similar. That is, within the giant structure of the Web, there are subsets of the Web that are themselves small bowties, and so on. The fractal nature of the web appears with respect to many of its properties including inlink, outlink, and PageRank power law exponents.

Recent work by Barabasi et al. [10, 11, 52] has uncovered the *scale-free* structure of the Web. This new discovery disputed earlier claims about the random network nature of the Web [45] and the small-world nature of the Web [114]. This model, called the scale-free model, describes well the various power law distributions that have been witnessed for node indegree, outdegree and PageRank as well as the average degree of separation [10, 49, 96]. The scale-free structure of the Web explains the emergence of hubs and a new node's increasing struggle to gain importance as time marches on. We view the use of the scale-free structure to improve PageRank computations as an uncharted area of future research.

Kamvar et al. [70] have considered the block domain structure of the Web to speed PageRank com-

putations. We predict other aggregation algorithms from numerical analysis, similar to their BlockRank algorithm, will play a greater role in the future, as researchers in Italy [21] have uncovered what appears to be a nearly completely decomposable structure [108] in the African Web.

The increase in *Intranet* search engines has driven other researchers to delineate the structural and philosophical differences between the WWW and Intranets [47]. The various intranets provide structure on yet another level and deserve greater attention.

Finally, we mention the level of structure considered by Bianchini et al. [19]. They examine the PageRank within a community of nodes. How do changes within the community affect the PageRank of community pages? How do changes outside the community affect the PageRank of community pages? How do changes inside the community affect the global PageRank? This provides for an interesting type of sensitivity analysis, with respect to groups of pages. In general, we believe that algorithms designed for the PageRank problem and tailored to exploit the various levels of structure on the Web should create significant improvements.

It is also worth noting that the ideas in this paper, concerning PageRank, extend to any network where finding the importance ranking of nodes is desired. For example, social networks, networks modeling the spread of disease, economic networks, citation networks, relational database networks, the Internet's network of routers, the email network, the power network and the transportation network. The book [10] by Barabasi contains an entertaining introduction to the science of networks, such as these.

10 Related Work

As alluded to in the introduction, HITS [74] is very similar to the PageRank model, but the differences are worth mentioning. Unlike PageRank, HITS is query-dependent due to its creation of a neighborhood graph of pages related to the query terms. HITS forms both an authority matrix and a hub matrix from the hyperlink adjacency matrix, rather than one Markov chain. As a result, HITS returns both authority and hub scores for each page, whereas PageRank returns only authority scores. PageRank is a global scoring vector, whereas HITS must compute two eigenvector calculations at query time. Numerous modifications and improvements to both HITS and PageRank and hybrids between the two have been created [4, 16, 17, 24, 30, 32, 37, 40, 41, 42, 43, 50, 51, 85, 99, 116]. Several groups have suggested incorporating text information into the link analysis [16, 38, 60, 67, 103]. Two other novel methods have been introduced, one based on entropy concepts [72] and another using flow [111]. A final related algorithm is the SALSA method of Lempel and Moran [81], which uses a bipartite graph of the Web to create two Markov chains for ranking pages.

Disclaimer We mention that PageRank is just one of many measures employed by Google to return relevant results to users. Many other heuristics are part of this successful engine; we have focused on only one. In addition, Google, of course, is very secretive about their technology. This survey paper, in no way, speaks for Google.

Acknowledgements We thank Cleve Moler for sharing his Mathworks dataset, `mathworks.dat`, and other web-crawling m-files. We also thank Ronny Lempel for providing us with several datasets that we used for testing. Finally, we thank the anonymous referee for the many valuable comments that improved the paper.

References

[1] PageRank explained. Web Rank Info. Optimize your rankings at Google! `http://www.webrankinfo.com/english/pagerank`, 2003.

[2] Why does my page's rank keep changing? Google PageRank information. `http://www.google.com/webmasters/4.html`, 2003.

[3] Serge Abiteboul, Mihai Preda, and Gregory Cobena. Adaptive on-line page importance computation. In *The Twelfth International World Wide Web Conference*, 2003.

[4] Dimitris Achlioptas, Amos Fiat, Anna R. Karlin, and Frank McSherry. Web search via hub synthesis. In *IEEE Symposium on Foundations of Computer Science*, pages 500–509, 2001.

[5] David Aldous. Random walks on finite groups and rapidly mixing Markov chains. In A. Dold and B. Eckmann, editors, *Lecture Notes in Mathematics*, volume 986, pages 243–297. Springer-Verlag, 1983.

[6] Arvind Arasu, Junghoo Cho, Hector Garcia-Molina, Andreas Paepcke, and Sriram Raghavan. Searching the Web. *ACM Transactions on Internet Technology*, 2001.

[7] Arvind Arasu, Jasmine Novak, Andrew Tomkins, and John Tomlin. PageRank computation and the structure of the Web: experiments and algorithms. In *The Eleventh International WWW Conference*, New York, May 2002. ACM Press.

[8] Konstantin Avrachenkov and Nelly Litvak. Decomposition of the google pagerank and optimal linking strategy. Technical report, INRIA, January 2004.

[9] Ricardo Baeza-Yates and Emilio Davis. Web page ranking using link attributes. In *The Thirteenth International World Wide Web Conference*, pages 328–329, New York, NY, USA, 2004. poster.

[10] Albert-Laszlo Barabasi. *Linked: The New Science of Networks*. Plume, 2003.

[11] Albert-Laszlo Barabasi, Reka Albert, and Hawoong Jeong. Scale-free characteristics of random networks: the topology of the world-wide web. *Physica A*, 281:69–77, 2000.

[12] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition*. SIAM, Philadelphia, PA, 1994.

[13] Luiz Andre Barroso, Jeffrey Dean, and Urs Holzle. Web search for a planet: The Google cluster architecture. *IEEE Micro*, pages 22–28, 2003.

[14] Abraham Berman and Robert J. Plemmons. *Nonnegative Matrices in the Mathematical Sciences*. Academic Press, Inc., 1979.

[15] Krishna Bharat, Andrei Broder, Monika Henzinger, Puneet Kumar, and Suresh Venkatasubramanian. The connectivity server: Fast access to linkage information on the Web. In *The Seventh World Wide Web Conference*, pages 469–477, Brisbane, Australia, 1998.

[16] Krishna Bharat and Monika R. Henzinger. Improved algorithms for topic distillation in hyperlinked environments. In *21st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 104–111, 1998.

[17] Krishna Bharat and George A. Mihaila. When experts agree: using non-affiliated experts to rank popular topics. *ACM Transactions on Information Systems*, 20(1):47–58, 2002.

[18] Monica Bianchini, Marco Gori, and Franco Scarselli. PageRank: A circuital analysis. In *The Eleventh International WWW Conference*, May 2002.

[19] Monica Bianchini, Marco Gori, and Franco Scarselli. Inside PageRank. *ACM Transactions on Internet Technology*, 5(1), 2005. To appear.

[20] Nancy Blachman, Eric Fredricksen, and Fritz Schneider. *How to Do Everything with Google*. McGraw-Hill, 2003.

[21] Paolo Boldi, Bruno Codenotti, Massimo Santini, and Sebastiano Vigna. Structural properties of the African web. In *The Eleventh International WWW Conference*, New York, May 2002. ACM Press.

[22] Paolo Boldi and Sebastiano Vigna. The WebGraph framework II: Codes for the World Wide Web. Technical Report 294-03, Universita di Milano, Dipartimento di Scienze dell' Informazione Engineering, 2003.

[23] Paolo Boldi and Sebastiano Vigna. The WebGraph framework I: Compression techniques. In *The Thirteenth International World Wide Web Conference*, pages 595–602, New York, NY, USA, 2004.

[24] Allan Borodin, Gareth O. Roberts, Jeffrey S. Rosenthal, and Panayiotis Tsaparas. Finding authorities and hubs from link structures on the World Wide Web. *World Wide Web*, pages 415–429, 2001.

[25] Sergey Brin, Rajeev Motwani, Lawrence Page, and Terry Winograd. What can you do with a Web in your pocket? *Data Engineering Bulletin*, 21:37–47, 1998.

[26] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 33:107–117, 1998.

[27] Sergey Brin, Lawrence Page, R. Motwami, and Terry Winograd. The PageRank citation ranking: bringing order to the Web. Technical Report 1999-0120, Computer Science Department, Stanford University, 1999.

[28] Andrei Broder, Ravi Kumar, and Marzin Maghoul. Graph structure in the Web. In *The Ninth International World Wide Web Conference*, pages 309–320, New York, May 2000. ACM Press.

[29] Andrei Broder, Ronny Lempel, Farzin Maghoul, and Jan Pedersen. Efficient PageRank approximation via graph aggregation. In *The Thirteenth International World Wide Web Conference*, pages poster, 484–485, New York, NY, USA, 2004.

[30] Soumen Chakrabarti, Byron Dom, David Gibson, Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew Tomkins. Spectral filtering for resource discovery. In *ACM SIGIR workshop on Hypertext Information Retrieval on the Web*, 1998.

[31] Yen-Yu Chen, Qingqing Gan, and Torsten Suel. I/O-efficient techniques for computing PageRank. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management (CIKM'02)*, pages 549–557, Virginia, USA, 2002.

[32] Zheng Chen, Jidong Wang, Liu Wenyin, and Wei-Ying Ma. A unified framework for web link analysis. In *Proceedings of Web Information Systems Engineering*, page 63, New York, 2002. ACM Press.

[33] Steve Chien, Cynthia Dwork, Ravi Kumar, and D. Sivakumar. Towards exploiting link evolution. In *Workshop on algorithms and models for the Web graph*, 2001.

[34] Grace E. Cho and Carl D. Meyer. Markov chain sensitivity measured by mean first passage times. *Linear Algebra and its Applications*, 313:21–28, 2000.

[35] Grace E. Cho and Carl D. Meyer. Comparison of perturbation bounds for the stationary distribution of a Markov chain. *Linear Algebra and its Applications*, 335(1–3):137–150, 2001.

[36] Junghoo Cho and Hector Garcia-Molina. The evolution of the web and implications for an incremental crawler. In *Proceedings of the Twenty-sixth International Conference on Very Large Databases*, pages 200–209, New York, 2000. ACM Press.

[37] David Cohn and Huan Chang. Learning to probabilistically identify authoritative documents. In *Proceedings of the 17th International Conference on Machine Learning*, pages 167–174, Stanford, CA, 2000.

[38] David Cohn and Thomas Hofmann. The missing link: a probabilistic model of document content and hyperlink connectivity. *Advances in Neural Information Processing Systems*, 13, 2001.

[39] Phil Craven. Google's PageRank explained. Web Workshop. `http://www.webworkshop.net/`, 2003.

[40] Brian D. Davison, Apostolos Gerasoulis, Konstantinos Kleisouris, Yingfang Lu, Hyun ju Seo, Wei Wang, and Baohua Wu. DiscoWeb: applying link analysis to web search. In *Eighth International World Wide Web Conference*, pages 148–149, Toronto, Canada, May 1999.

[41] Michelangelo Diligenti, Marco Gori, and Marco Maggini. Web page scoring systems for horizontal and vertical search. In *The Eleventh International World Wide Web Conference*, pages 508–516, Honolulu, Hawaii, USA, 2002. ACM Press.

[42] Chris Ding, Xiaofeng He, Parry Husbands, Hongyuan Zha, and Horst Simon. Link analysis: hubs and authorities on the World Wide Web. Technical Report 47847, Lawrence Berkeley National Laboratory, May 2001.

[43] Chris Ding, Xiaofeng He, Hongyuan Zha, and Horst Simon. PageRank, HITS and a unified framework for link analysis. In *Proceedings of the 25th ACM SIGIR Conference*, pages 353–354, Tampere, Finland, August 2002.

[44] Cynthia Dwork, Ravi Kumar, and Moni Naor and D. Sivakumar. Rank aggregation methods for the Web. In *The Tenth International World Wide Web Conference*, 2001.

[45] Paul Erdos and Alfred Renyi. On random graphs I. *Math. Debrechen*, 6:290–297, 1959.

[46] Ronald Fagin, Anna R. Karlin, Jon Kleinberg, Prabhakar Raghavan, Sridhar Rajagopalan, Ronitt Rubinfeld, Madhu Sudan, and Andrew Tomkins. Random walks with 'back buttons'. In *32nd ACM Symposium on Theory of Computing*, 2000.

[47] Ronald Fagin, Ravi Kumar, Kevin S. McCurley, Jasmine Novak, D. Sivakumar, John A. Tomlin, and David P. Williamson. Searching the workplace web. In *The Twelfth International World Wide Web Conference*, pages 366–375, New York, 2003. ACM Press.

[48] Ronald Fagin, Ravi Kumar, and D. Sivakumar. Comparing top k lists. In *ACM SIAM Symposium on Discrete Algorithms*, pages 28–36, 2003.

[49] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *SIGCOMM*, pages 251–262, 1999.

[50] Ayman Farahat, Thomas Lofaro, Joel C. Miller, Gregory Rae, F. Schaefer, and Lesley A. Ward. Modifications of Kleinberg's HITS algorithm using matrix exponentiation and web log records. In *ACM SIGIR Conference*, pages 444–445, September 2001.

[51] Ayman Farahat, Thomas Lofaro, Joel C. Miller, Gregory Rae, and Lesley A. Ward. Existence and uniqueness of ranking vectors for linear link analysis. *SIAM Journal on Scientific Computing*, 2004. submitted April 2004.

[52] Illes J. Farkas, Imre Derenyi, Albert-Laszlo Barabasi, and Tamas Vicsek. Spectra of real-world graphs: Beyond the semicircle law. *Physical Review E*, 64, 2001.

[53] Dennis Fetterly, Mark Manasse, Marc Najork, and Janet L. Wiener. A large-scale study of the evolution of web pages. In *The Twelfth International World Wide Web Conference*, 2003.

[54] Robert E. Funderlic and Carl D. Meyer. Sensitivity of the stationary distribution vector for an ergodic Markov chain. *Linear Algebra and its Applications*, 76:1–17, 1986.

[55] Robert E. Funderlic and Robert J. Plemmons. Updating **LU** factorizations for computing stationary distributions. *SIAM Journal on Algebraic and Discrete Methods*, 7(1):30–42, 1986.

[56] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The Google file system. In *Proceedings of the Nineteenth ACM symposium on Operating Systems principles*, pages 29–43, New York, USA, 2003.

[57] Gene H. Golub and Carl D. Meyer. Using the QR factorization and group inverse to compute, differentiate and estimate the sensitivity of stationary probabilities for Markov chains. *SIAM Journal on Algebraic and Discrete Methods*, 17:273–281, 1986.

[58] Taher H. Haveliwala. Efficient computation of PageRank. Technical Report 1999-31, Computer Science Department, Stanford University, 1999.

[59] Taher H. Haveliwala. Efficient encodings for document ranking vectors. Technical report, Computer Science Department, Stanford University, November 2002.

[60] Taher H. Haveliwala. Topic-sensitive PageRank. In *The Eleventh International WWW Conference*, May 2002.

[61] Taher H. Haveliwala and Sepandar D. Kamvar. The second eigenvalue of the Google matrix. Technical Report 2003-20, Stanford University, 2003.

[62] Taher H. Haveliwala, Sepandar D. Kamvar, and Glen Jeh. An analytical comparison of approaches to personalizing PageRank. Technical report, Stanford University, 2003.

[63] Monika Henzinger. Algorithmic challenges in web search engines. *Journal of Internet Mathematics*, 1(1):115–126, 2003.

[64] Monika Henzinger, Bay-Wei Chang, Brian Milch, and Sergey Brin. Query-free news search. In *The Twelfth International World Wide Web Conference*, 2003.

[65] Monika Henzinger, Rajeev Motwani, and Craig Silverstein. Challenges in web search engines. In *SIGIR Forum*, 2002.

[66] Ilse C. F. Ipsen and Carl D. Meyer. Uniform stability of Markov chains. *SIAM Journal on Matrix Analysis and Applications*, 15(4):1061–1074, 1994.

[67] Glen Jeh and Jennifer Widom. Scaling personalized web search. Technical report, Stanford University, 2002.

[68] Sepandar D. Kamvar and Taher H. Haveliwala. The condition number of the PageRank problem. Technical report, Stanford University, 2003.

[69] Sepandar D. Kamvar, Taher H. Haveliwala, and Gene H. Golub. Adaptive methods for the computation of PageRank. Technical Report 2003-26, Stanford University, 2003.

[70] Sepandar D. Kamvar, Taher H. Haveliwala, Christopher D. Manning, and Gene H. Golub. Exploiting the block structure of the Web for computing PageRank. Technical Report 2003-17, Stanford University, 2003.

[71] Sepandar D. Kamvar, Taher H. Haveliwala, Christopher D. Manning, and Gene H. Golub. Extrapolation methods for accelerating PageRank computations. Twelfth International World Wide Web Conference, 2003.

[72] Hung-Yu Kao, Ming-Syan Chen, Shian-Hua Lin, and Jan-Ming Ho. Entropy-based link analysis for mining web informative structures. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 574–581, McLean, Virginia, USA, 2002. ACM Press.

[73] John G. Kemeny and Laurie J. Snell. *Finite Markov Chains*. D. Van Nostrand, New York, 1960.

[74] Jon Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46, 1999.

[75] Amy Langville and Carl Meyer. A reordering for the PageRank problem. Technical Report CRSC Tech Report crsc-tr04-06, Center For Research in Scientific Computation, 2004. submitted to SIAM Journal on Scientific Computing, 2004.

[76] Amy N. Langville and Carl D. Meyer. Updating PageRank using the group inverse and stochastic complementation. Technical Report crsc02-tr32, North Carolina State University, Mathematics Department, CRSC, 2002.

[77] Amy N. Langville and Carl D. Meyer. Updating the stationary vector of an irreducible Markov chain. Technical Report crsc02-tr33, N. C. State, Mathematics Dept., CRSC, 2002.

[78] Amy N. Langville and Carl D. Meyer. A survey of eigenvector methods of web information retrieval. *The SIAM Review*, 2004. Accepted in October 2004.

[79] Chris Pan-Chi Lee, Gene H. Golub, and Stefanos A. Zenios. A fast two-stage algorithm for computing PageRank and its extensions. Technical Report SCCM-2003-15, Scientific Computation and Computational Mathematics, Stanford University, 2003.

[80] Hyun Chul Lee and Allan Borodin. Perturbation of the hyperlinked environment. In *Lecture Notes in Computer Science: Proceedings of the Ninth International Computing and Combinatorics Conference*, volume 2697, Heidelberg, 2003. Springer-Verlag.

[81] Ronny Lempel and S. Moran. The stochastic approach for link-structure analysis (SALSA) and the TKC effect. In *The Ninth International WWW Conference*, May 2000.

[82] Ronny Lempel and Shlomo Moran. Rank-stability and rank-similarity of link-based web ranking algorithms in authority-connected graphs. In *Second Workshop on Algorithms and Models for the Web-Graph (WAW 2003)*, Budapest, Hungary, May 2003.

[83] Yizhou Lu, Benyu Zhang, Wensi Xi, Zheng Chen, Yi Liu, Michael R. Lyu, and Wei-Ying Ma. The PowerRank Web link analysis algorithm. In *The Thirteenth International World Wide Web Conference*, pages 254–255, New York, NY, USA, 2004. poster.

[84] Fabien Mathieu and Mohamed Bouklit. The effect of the back button in a random walk: Application for PageRank. In *The Thirteenth International World Wide Web Conference*, pages poster, 370–371, New York, NY, USA, 2004.

[85] Alberto O. Mendelzon and Davood Rafiei. What do the neighbours think? Computing web page reputations. *IEEE Data Engineering Bulletin*, 23(3):9–16, 2000.

[86] Alberto O. Mendelzon and Davood Rafiei. An autonomous page ranking method for metasearch engines. In *The Eleventh International WWW Conference*, May 2002.

[87] Carl D. Meyer. The character of a finite Markov chain. *Linear Algebra, Markov Chains, and Queueing Models, IMA Volumes in Mathematics and its Applications, Ed., C. D. Meyer and R. J. Plemmons, Springer-Verlag*, 48:47–58, 1993.

[88] Carl D. Meyer. Sensitivity of the stationary distribution of a Markov chain. *SIAM Journal on Matrix Analysis and Applications*, 15(3):715–728, 1994.

[89] Carl D. Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, Philadelphia, 2000.

[90] Carl D. Meyer and James M. Shoaf. Updating finite Markov chains by using techniques of group matrix inversion. *Journal of Statistical Computation and Simulation*, 11:163–181, 1980.

[91] Carl D. Meyer and G. W. Stewart. Derivatives and perturbations of eigenvectors. *SIAM Journal on Numerical Analysis*, 25:679–691, 1988.

[92] Cleve Moler. The world's largest matrix computation. *Matlab News and Notes*, pages 12–13, October 2002.

[93] Cleve B. Moler. *Numerical Computing with MATLAB*. SIAM, 2004.

[94] Andrew Y. Ng, Alice X. Zheng, and Michael I. Jordan. Link analysis, eigenvectors and stability. In *The Seventh International Joint Conference on Artificial Intelligence*, 2001.

[95] Andrew Y. Ng, Alice X. Zheng, and Michael I. Jordan. Stable algorithms for link analysis. In *Proceedings of the 24th Annual International ACM SIGIR Conference*. ACM, 2001.

[96] Gopal Pandurangan, Prabhakar Raghavan, and Eli Upfal. Using PageRank to Characterize Web Structure. In *The Eighth Annual International Computing and Combinatorics Conference (COCOON)*, 2002.

[97] Luca Pretto. *Link Analysis Techniques for ranking webpages*. PhD thesis, University of Padua, 2002.

[98] Luca Pretto. A theoretical analysis of PageRank. In *Proceedings of the Ninth International Symposium on String Processing and Information Retrieval*, pages 131–144, Lisbon, Portugal, September 2002.

[99] Davood Rafiei and Alberto O. Mendelzon. What is this page known for? computing webpage reputations. In *The Ninth International WWW Conference*, pages 823–835. Elsevier Science, May 2000.

[100] Sriram Raghavan and Hector Garcia-Molina. Compressing the graph structure of the Web. In *Proceedings of the IEEE Conference on Data Compression*, pages 213–222, March 2001.

[101] Sriram Raghavan and Hector Garcia-Molina. Towards compressing web graphs. In *Proceedings of the IEEE Conference on Data Compression*, pages 203–212, March 2001.

[102] Sriram Raghavan and Hector Garcia-Molina. Representing web graphs. In *Proceedings of the 19th IEEE Conference on Data Engineering*, Bangalore, India, March 2003.

[103] Matthew Richardson and Petro Domingos. The intelligent surfer: probabilistic combination of link and content information in PageRank. *Advances in Neural Information Processing Systems*, 14:1441–1448, 2002.

[104] Chris Ridings. PageRank explained: everything you've always wanted to know about PageRank. online at `http://www.rankwrite.com/`. Accessed on May 22, 2002.

[105] Chris Ridings and Mike Shishigin. PageRank uncovered. online at `www.voelspriet2.nl/PageRank.pdf`. Accessed on September 19, 2002.

[106] Eugene Seneta. Sensivity analysis, ergodicity coefficients, and rank-one updates for finite Markov chains. In William J. Stewart, editor, *Numerical Solutions of Markov Chains*, pages 121–129, 1991.

[107] Chris Sherman. Teoma vs. google, round 2. *Silicon Valley Internet*, 2002. `http://dc.internet.com/news/print.php/1002061`.

[108] William J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.

[109] Marcin Sydow. Random surfer with back step. In *The Thirteenth International World Wide Web Conference*, pages poster, 352–353, New York, NY, USA, 2004.

[110] Kristen Thorson. Modeling the Web and the computation of PageRank. Undergraduate thesis, Hollins University, 2004.

[111] John A. Tomlin. A new paradigm for ranking pages on the World Wide Web. In *The Twelfth International World Wide Web Conference*, 2003.

[112] Michael Totty and Mylene Mangalindan. As google becomes web's gatekeeper, sites fight to get in. *Wall Street Journal*, CCXLI(39), 2003. February 26.

[113] Ah Chung Tsoi, Gianni Morini, Franco Scarselli, and Markus Hagenbuchner. Adaptive ranking of webpages. In *The Twelfth International World Wide Web Conference*, 2003.

[114] Duncan J. Watts. *Small Worlds*. Princeton University Press, 1999.

[115] James H. Wilkinson. *The Algebraic Eigenvalue Problem*. Claredon Press, 1965.

[116] Dell Zhang and Yisheng Dong. An efficient algorithm to rank web resources. *Computer Networks*, 33:449–455, 2000.